The String-to-String Correction Problem

ROBERT A. WAGNER

Vanderbilt University, Nashville, Tennessee

AND

MICHAEL J. FISCHER

Massachusetts Institute of Technology, Cambridge, Massachusetts

ABSTRACT. The *string-to-string correction problem* is to determine the distance between two strings as measured by the minimum cost sequence of "edit operations" needed to change the one string into the other. The edit operations investigated allow changing one symbol of a string into another single symbol, deleting one symbol from a string, or inserting a single symbol into a string. An algorithm is presented which solves this problem in time proportional to the product of the lengths of the two strings. Possible applications are to the problems of automatic spelling correction and determining the longest subsequence of characters common to two strings.

KEY WORDS AND PHRASES: string correction, editing, string modification, correction, spelling correction, longest common subsequence

CR CATEGORIES: 3.79, 4.12, 4.22, 5.23, 5.25

1. *Introduction*

Morgan [1] considers four editing operations which can be applied to keypunched words in order to undo certain common keypunch errors. His paper describes a technique for finding those language tokens (usually complier key words, such as BEGIN or WRITE) which lie a distance of one edit operation away from the given, presumably incorrect, input token.

Based on three of Morgan's operations, we define a general notion of "distance" between two strings and present an algorithm for computing the distance in time proportional to the product of the lengths of the strings. The operations we consider are: (1) changing one character to another single character; (2) deleting one character from the given string; (3) inserting a single character into the given string.

This notion of edit distance and the efficient algorithm for computing it have obvious applications to problems of spelling correction and may be useful in choosing mutually distant key words in the design of a programming language. The algorithm may also be

The work reported herein was supported in part by the National Science Foundation under Grants GJ-28176 to Cornell University, GJ-33014 to Vanderbilt University, and GJ-34671 to MIT Project MAC, and in part by the Artificial Intelligence Laboratory, an MIT research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research contract number N00014-70-A-0362-0003.

Authors' addresses: Robert A. Wagner, Systems and Information Science Department, Vanderbilt University, Nashville, TN 37235; Michael J. Fischer, Project MAC, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139.

used, as a special case, to find the longest subsequence of characters common to two strings.

2. *Edit Distance*

Let A be a finite string (or sequence) of characters (or symbols). $A\langle i\rangle$ is the ith character of string A; $A\langle i:j\rangle$ is the ith through jth characters (inclusive) of A (so $A\langle i:j\rangle = A\langle i\rangle A\langle i+1\rangle \cdots A\langle j\rangle$), and $A\langle i:j\rangle = \Lambda$, the null string, if $i > j$. $|A|$ denotes the length (number of characters) of string A.

An *edit operation* is a pair $(a, b) \neq (\Lambda, \Lambda)$ of strings of length less than or equal to 1 and is usually written $a \to b$. String B results from the application of the operation $a \to b$ to string A, written $A \Rightarrow B$ *via* $a \to b$, if $A = \sigma a \tau$ and $B = \sigma b \tau$ for some strings σ and τ. (Readers familiar with formal language theory will note the similarity between an edit operation and a production of a grammar.) We call $a \to b$ a *change* operation if $a \neq \Lambda$ and $b \neq \Lambda$; a *delete* operation if $b = \Lambda$; and an *insert* operation if $a = \Lambda$.

Let S be a sequence $s_1, s_2, \cdots, s_m$ of edit operations (or *edit sequence* for short). An *S-derivation from A to B* is a sequence of strings $A_0, A_1, \cdots, A_m$ such that $A = A_0$, $B = A_m$, and $A_{i-1} \Rightarrow A_i$ via s_i for $1 \leq i \leq m$. We say S *takes A to B* if there is some S-derivation from A to B.

Now let γ be an arbitrary cost function which assigns to each edit operation $a \to b$ a nonnegative real number $\gamma(a \to b)$. Extend γ to a sequence of edit operations $S = s_1, s_2, \cdots, s_m$ by letting $\gamma(S) = \sum_{i=1}^{m} \gamma(s_i)$. (If $m = 0$, we define $\gamma(S) = 0$.) We now let the *edit distance* $\delta(A, B)$ from string A to string B be the minimum cost of all sequences of edit operations which transform A into B. Formally, $\delta(A, B) = \min\{\gamma(S) \mid S$ is an edit sequence taking A to $B\}$.

We will assume henceforth that $\gamma(a \to b) = \delta(a, b)$ for all edit operations $a \to b$. (Equivalently, we may assume that $\gamma(a \to a) = 0$ and $\gamma(a \to b) + \gamma(b \to c) \geq \gamma(a \to c)$.) This leads to no loss of generality with respect to the class of distance functions we are considering, for if δ is the distance function associated with a cost function γ, it is easily verified that δ is also the distance function associated with the cost function γ' defined by $\gamma'(a \to b) = \delta(a, b)$, and γ' has the desired property.

Note that if δ were symmetric and strictly positive on each edit operation $a \to b$ for which $a \neq b$, then γ would be a metric on the space of all strings—hence our use of the term "distance." We remark also that cost functions which depend on the particular characters affected by an edit operation might be useful in spelling correction, where for example because of the conventional keyboard arrangement it may be far more likely that a character "*A*" be mistyped as an "*S*" than as a "*Y*."

3. *Traces*

To simplify our problem of finding the edit distance between two strings A and B, we define a cost function on some structures called *traces* and show that traces have the properties:

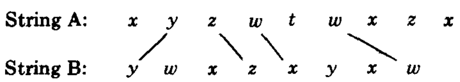
(P1) for every trace T from A to B, there is an edit sequence S taking A to B such that $\gamma(S) = \text{cost}(T)$;

(P2) for every edit sequence S taking A to B, there is a trace T from A to B such that $\text{cost}(T) \leq \gamma(S)$.

Thus, $\delta(A, B)$ is equal to the minimum cost trace from A to B, so we will be able to confine our attention to finding minimum cost traces.

Intuitively, a trace is a description of how an edit sequence S transforms A into B but ignoring the order in which things happen and any redundancy in S.

Consider the diagram:



A line in this diagram joining character position i of A to position j of B means that $B\langle j\rangle$ is derived from $A\langle i\rangle$, either directly if $A\langle i\rangle = B\langle j\rangle$ and S leaves $A\langle i\rangle$ unchanged or indirectly if S applies one or more change operations to $A\langle i\rangle$. Certain character positions of A are untouched by lines in our diagram; these positions represent characters of A deleted by S (either directly or perhaps as the result of one or more change operations followed by a delete). Similarly, certain positions of B are untouched by lines; these positions represent characters inserted into A by S.

Formally, a *trace from A to B* (or *trace* when the strings A and B are understood) is a triple (T, A, B), where T is any set of ordered pairs of integers (i, j) satisfying:

(1) $1 \le i \le |A|$ and $1 \le j \le |B|$;

(2) for any two distinct pairs (i_1, j_1) and (i_2, j_2) in T, (a) $i_1 \ne i_2$ and $j_1 \ne j_2$; (b) $i_1 < i_2$ iff $j_1 < j_2$.

A pair (i, j) describes a line joining position i of A to position j of B, and we say (i, j) *touches* those positions. Condition (1) ensures that our lines actually touch character positions of the respective strings. Condition (2a) ensures that each character position of either string is touched by at most one line; condition (2b) ensures that no two lines cross. Where there is no confusion, we will not distinguish between the triple (T, A, B) and the set of pairs T.

Let T be a trace from A to B. Let I and J be the sets of positions in A and B respectively not touched by any line in T. We define the cost of T:

$$\text{cost}(T) = \sum_{(i,j)\in T} \gamma(A\langle i\rangle \to B\langle j\rangle) + \sum_{i\in I} \gamma(A\langle i\rangle \to \Lambda) + \sum_{j\in J} \gamma(\Lambda \to B\langle j\rangle).$$

Thus, the cost of T is just the cost of the edit sequence S taking A to B which consists of a change instruction $A\langle i\rangle \to B\langle j\rangle$ for each pair $(i, j) \in T$, a delete instruction $A\langle i\rangle \to \Lambda$ for every position i in A not touched by a line in T, and an insert instruction $\Lambda \to B\langle j\rangle$ for every position j in B not touched by a line in T. Hence, property (P1) of traces follows.

Traces may be composed. Let T_1 be a trace from A to B and let T_2 be a trace from B to C. It is readily verified that $T = T_1 \circ T_2$ is a trace from A to C, where $\circ$ denotes ordinary composition of relations.[1]

LEMMA 1. *$Cost(T_1 \circ T_2) \le cost(T_1) + cost(T_2)$, where T_1 is a trace from A to B and T_2 is a trace from B to C.*

The proof relies on our assumption that $\gamma(a \to b) = \delta(a, b)$ and is omitted.

To verify that property (P2) holds for traces, we show by induction on m that if $S = s_1, s_2, \cdots, s_m$ is a sequence of edit operations and $(A_0, A_1, \cdots, A_m)$ is an S-derivation (from A_0 to A_m), then there is a trace T from A_0 to A_m such that $\text{cost}(T) \le \gamma(S)$.

If $m = 0$, let $T = \{(i, i) \mid 1 \le i \le |A_0|\}$ be a trace from A_0 to A_0. Then $\text{cost}(T) = 0 = \gamma(S)$ and the induction hypothesis holds.

If $m > 0$, by induction, there is a trace T_1 from A_0 to A_{m-1} such that $\text{cost}(T_1) \le \gamma(s_1, \cdots, s_{m-1})$. $A_{m-1} \Rightarrow A_m$ via $s_m = a \to b$, so there are strings σ and τ such that $A_{m-1} = \sigma a \tau$ and $A_m = \sigma b \tau$. Let T_2 be the trace from A_{m-1} to A_m defined by

$$T_2 = \{(i, i) \mid 1 \le i \le |\sigma|\} \cup \{(i, i + d) \mid |\sigma a| + 1 \le i \le |A_{m-1}|\} \cup L,$$

where $d = |b| - |a| \in \{-1, 0, 1\}$ and

$$L = \begin{cases} \{(|\sigma| + 1, |\sigma| + 1)\} & \text{if } s_m \text{ is a change instruction;} \\ \varnothing & \text{otherwise.} \end{cases}$$

Clearly, T_2 is a trace and $\text{cost}(T_2) = \gamma(a \to b) = \gamma(s_m)$.

Now let $T = T_1 \circ T_2$. T is a trace from A_0 to A_m. By Lemma 1,

$$\text{cost}(T) \le \text{cost}(T_1) + \text{cost}(T_2) \le \gamma(s_1, \cdots, s_{m-1}) + \gamma(s_m) = \gamma(S),$$

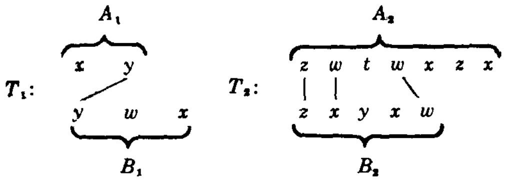
so property (P2) holds for S. By induction, it holds for all sequences S.

[1] $T_1 \circ T_2 = \{(i, j) \mid (i, k) \in T_1$ and $(k, j) \in T_2$ for some $k\}$.

From properties (P1) and (P2) of traces, we have:
THEOREM 1. $\delta(A, B) = min\{cost(T) \mid T \text{ is a trace from } A \text{ to } B\}$.

4. *Computation of Edit Distance*

Now return to the diagrammatic representation of a trace T from A to B. Let $A = A_1A_2$, $B = B_1B_2$, and suppose no line of T connects a character of A_i to a character of B_j for $i \neq j$, $i, j \in \{1, 2\}$. Then a trace (T, A, B) can be split into two traces (T_1, A_1, B_1) and (T_2, A_2, B_2) as illustrated.



Furthermore, $\text{cost}(T) = \text{cost}(T_1) + \text{cost}(T_2)$, so if T is a least cost trace from A to B, then T_i is a least cost trace from A_i to B_i, $i \in \{1, 2\}$.

Every trace T from A to B can in fact be split into two traces T_1 and T_2 as above such that the lengths of A_2 and B_2 are each at most one but they are not both zero. This is the key idea for the following theorem, upon which the edit distance algorithm is based.

Notation. Let A and B be strings. Define $A(i) = A\langle 1 : i\rangle$, $B(j) = B\langle 1 : j\rangle$, and $D(i, j) = \delta(A(i), B(j))$, $0 \leq i \leq |A|$, $0 \leq j \leq |B|$. We note that by Theorem 1, $D(i, j)$ is also the cost of the least cost trace from $A(i)$ to $B(j)$.

THEOREM 2.

$$\begin{aligned} D(i, j) = min\{&D(i-1, j-1) + \gamma(A\langle i\rangle \rightarrow B\langle j\rangle), \\ &D(i-1, j) + \gamma(A\langle i\rangle \rightarrow \Lambda), \\ &D(i, j-1) + \gamma(\Lambda \rightarrow B\langle j\rangle)\} \end{aligned}$$

for all i, j, $1 \leq i \leq |A|$, $1 \leq j \leq |B|$.

PROOF. Let T be a least cost trace from $A(i)$ to $B(j)$. If $A\langle i\rangle$ and $B\langle j\rangle$ are both touched by lines in T, they must both be touched by the same line, since otherwise these lines in T would cross. Then at least one of the following three cases must hold:

Case 1. $A\langle i\rangle$ and $B\langle j\rangle$ are joined by a line of T (i.e. $(i, j) \in T$). Then the cost of T is $m_1 = D(i-1, j-1) + \gamma(A\langle i\rangle \rightarrow B\langle j\rangle)$, corresponding to the cost of transforming $A(i-1)$ to $B(j-1)$ plus the cost of changing $A\langle i\rangle$ to $B\langle j\rangle$.

Case 2. $A\langle i\rangle$ is not touched by any line in T. Then the cost of T is $m_2 = D(i-1, j) + \gamma(A\langle i\rangle \rightarrow \Lambda)$, corresponding to the costs of transforming $A(i-1)$ to $B(j)$ and deleting $A\langle i\rangle$.

Case 3. $B\langle j\rangle$ is not touched by any line in T. Then the cost of T is $m_3 = D(i, j-1) + \gamma(\Lambda \rightarrow B\langle j\rangle)$, corresponding to the costs of transforming $A(i)$ to $B(j-1)$ and inserting character $B\langle j\rangle$.

Since one of the three cases above must hold and $D(i, j)$ is to be a minimum, $D(i, j) = \min(m_1, m_2, m_3)$. □

THEOREM 3. $D(0, 0) = 0$; $D(i, 0) = \sum_{r=1}^{i} \gamma(A\langle r\rangle \rightarrow \Lambda)$; *and* $D(0, j) = \sum_{r=1}^{j} \gamma(\Lambda \rightarrow B\langle r\rangle)$, $1 \leq i \leq |A|$ *and* $1 \leq j \leq |B|$.

PROOF. The only (and hence least cost) trace from $A(i)$ to $B(j)$ when either i or $j = 0$ is $\varnothing$, and hence no lines touch $A(i)$ or $B(j)$. The theorem follows immediately from the definition of the cost of a trace. □

Theorems 2 and 3 justify that Algorithm X (below) correctly computes $D(i, j)$ for $0 \leq i \leq |A|$ and $0 \leq j \leq |B|$.

AGLORITHM X

1. $D[0, 0] := 0$;
2. **for** $i := 1$ **to** $|A|$ **do** $D[i, 0] := D[i-1, 0] + \gamma(A\langle i\rangle \rightarrow \Lambda)$;

```
for j := 1 to |B| do D[0, j] := D[0, j − 1] + γ(Λ → B⟨j⟩);
for i := 1 to |A| do
  for j := 1 to |B| do begin
    m1 := D[i − 1, j − 1] + γ(A⟨i⟩ → B⟨j⟩);
    m2 := D[i − 1, j] + γ(A⟨i⟩ → Λ);
    m3 := D[i, j − 1] + γ(Λ → B⟨j⟩);
    D[i, j] := min(m1, m2, m3);
    end;
```

By inspection, we see that the total amount of time used by Algorithm X is proportional to the number of assignment statements executed (exclusive of those implicit in the **for**-loops). This number is exactly $1 + |A| + |B| + 4 \times |A| \times |B|$, so the total time is $O(|A| \times |B|)$.

If an actual trace T from A to B of least cost is desired, Algorithm Y will print the pairs in T using only the information stored in array D by Algorithm X.

ALGORITHM Y

```
1. i := |A|; j := |B|;
2. while(i ≠ 0 & j ≠ 0) do
3.   if D[i, j] = D[i − 1, j] + γ(A⟨i⟩ → Λ) then i := i − 1;
4.   else if D[i, j] = D[i, j − 1] + γ(Λ → B⟨j⟩) then j := j − 1;
5.   else begin
6.     print((i, j));
7.     i := i − 1; j := j − 1;
8.     end;
```

In order to prove that Algorithm Y works correctly, we consider for every pair of natural numbers I and J the behavior of the algorithm when started at step 2 with variables i and j initialized to I and J respectively. Let $T(I, J)$ be the set of pairs printed by the algorithm if the execution eventually terminates, and $T(I, J)$ is undefined otherwise.

THEOREM 4. *If* $0 \leq I \leq |A|$ *and* $0 \leq J \leq |B|$, *then* $T(I, J)$ *is defined*, $\mathbf{T} = (T(I, J), A(I), B(J))$ *is a trace, and cost* $(\mathbf{T}) = D(I, J)$.

PROOF. We proceed by induction on the sum $I + J$.

The theorem is vacuously true for $I + J < 0$.

Now let $r \geq 0$ and suppose the theorem holds for all I', J' such that $I' + J' < r$. Let $I + J = r$. If either I or J is 0, step 2 terminates immediately and $T(I, J) = \varnothing$ is the only trace from $A(I)$ to $B(J)$; hence its cost is minimal. If neither I nor J is zero, we have three cases:

Case 1. The test in step 3 succeeds. Then $D(I, J) = D(I - 1, J) + \gamma(A\langle I\rangle \to \Lambda)$. The algorithm then proceeds by decrementing i and returning to step 2. Variable i now has the value $I - 1$, and j is unchanged. By induction, $T(I - 1, J)$ is defined, and $\mathbf{T} = (T(I - 1, J), A(I - 1), B(J))$ is a trace of cost $D(I - 1, J)$. No output was produced before returning to step 2,.so $T(I, J) = T(I - 1, J)$, and $\mathbf{T}' = (T(I, J), A(I), B(J))$ is a trace. Then

$$\text{cost}(\mathbf{T}') = \text{cost}(\mathbf{T}) + \gamma(A\langle I\rangle \to \Lambda) = D(I - 1, J) + \gamma(A\langle I\rangle \to \Lambda) = D(I, J).$$

Case 2. The test in step 3 fails but the one in step 4 succeeds. The proof for this case is exactly analogous to case 1.

Case 3. The tests in steps 3 and 4 both fail. Hence $D(I, J) \neq D(I - 1, J) + \gamma(A\langle I\rangle \to \Lambda)$ and $D(I, J) \neq D(I, J - 1) + \gamma(\Lambda \to B\langle J\rangle)$. By Theorem 2, it must be the case that $D(I, J) = D(I - 1, J - 1) + \gamma(A\langle I\rangle \to B\langle J\rangle)$.

The block from steps 5–8 is then executed. This causes the pair (I, J) to be printed, and when step 2 is reentered, both i and j have been decremented. By induction, $T(I - 1, J - 1)$ is defined, and $\mathbf{T} = (T(I - 1, J - 1), A(I - 1), B(I - 1))$ is a trace of cost $D(I - 1, J - 1)$. Hence, $T(I, J) = \{(I, J)\} \cup T(I - 1, J - 1)$, and $\mathbf{T}' = (T(I, J),$

$A(I), B(J))$ is a trace. Then

$$\text{cost}(\mathbf{T}') = \text{cost}(\mathbf{T}) + \gamma(A\langle I\rangle \rightarrow B\langle J\rangle) = D(I-1, J-1) + \gamma(A\langle I\rangle \rightarrow B\langle J\rangle)$$
$$= D(I, J).$$

Hence, in all three cases, the theorem holds for I and J. By induction, the theorem holds for all I and J. □

Algorithm Y when started at the beginning first enters step 2 with $i = |A|$ and $j = |B|$. By Theorem 4, it eventually terminates and prints the pairs in $T(|A|, |B|)$, which is a least cost trace from A to B as desired.

We note that in all three cases of the proof of Theorem 4, either i or j (or both) is decremented, and Algorithm Y terminates when either reaches 0. Hence, the loop is executed at most $|A| + |B|$ times, so the total running time of Algorithm Y is $O(|A| + |B|)$.

5. *Longest Common Subsequences*

Let U and V be strings. U is a *subsequence* of V if there exist integers $1 \leq r_1 < r_2 < \cdots < r_n \leq |V|$ such that $U\langle i\rangle = V\langle r_i\rangle$, $1 \leq i \leq n = |U|$. Given two strings A and B, U is a *common subsequence* of A and B if U is a subsequence of both A and B.

Let $\rho(A, B)$ be the length of the longest common subsequence of A and B. It is immediate from the definition of a trace that $\rho(A, B)$ is also the maximum number of pairs (i, j) in any trace from A to B for which $A\langle i\rangle = B\langle j\rangle$. Let T be such a trace.

Define γ so that the cost of an insert or a delete operation is 1, and let the cost of a change operation $a \rightarrow b$ be 0 if $a = b$ and 2 if $a \neq b$. Under this cost assignment, $\text{cost}(T) = |A| + |B| - 2\rho(A, B)$. T is a least cost trace from A to B, so $\text{cost}(T) = \delta(A, B)$. Hence, $\rho(A, B) = (|A| + |B| - \delta(A, B))/2$ can be computed in time $O(|A| \times |B|)$ using Algorithm X. The longest common subsequence itself can be found easily from T which in turn can be obtained using Algorithm Y.

ACKNOWLEDGMENT. The authors are grateful to M. Paterson and V. Pratt for many helpful discussions and to A. Meyer for a critical reading of a draft of this paper.

REFERENCE

1. MORGAN, H. L. Spelling correction in systems programs. *Comm. ACM 13*, 2 (Feb. 1970), 90–94.

RECEIVED FEBRUARY 1972; REVISED MARCH 1973

U. S. POSTAL SERVICE

STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION

(Act of August 12, 1970: Section 3685, Title 39, United States Code)

SEE INSTRUCTIONS ON PAGE 2 (REVERSE)

1. TITLE OF PUBLICATION
Journal of the Association for Computing Machinery

2. DATE OF FILING
Oct. 1, 1973

3. FREQUENCY OF ISSUE
Quarterly

4. LOCATION OF KNOWN OFFICE OF PUBLICATION *(Street, city, county, state, ZIP code) (Not printers)*
1133 Avenue of the Americas, New York, N.Y. 10036

5. LOCATION OF THE HEADQUARTERS OR GENERAL BUSINESS OFFICES OF THE PUBLISHERS *(Not printers)*
1133 Avenue of the Americas, New York, N.Y. 10036

6. NAMES AND ADDRESSES OF PUBLISHER, EDITOR, AND MANAGING EDITOR

PUBLISHER *(Name and address)*
Association for Computing Machinery Inc., 1133 Avenue of the Americas, New York, N.Y. 10036

EDITOR *(Name and address)*
Raymond E. Miller, IBM Thomas J. Watson Research Ctr., P.O. Box 218, Yorktown Heights, N.Y. 10598

MANAGER EDITOR *(Name and address)*
Myrtle R. Kellington, 1133 Avenue of the Americas, New York, N.Y. 10036

7. OWNER *(If owned by a corporation, its name and address must be stated and also immediately thereunder the names and addresses of stockholders owning or holding 1 percent or more of total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address, as well as that of each individual must be given.)*

NAME	ADDRESS
Association for Computing Machinery Inc.	1133 Avenue of the Americas New York, N.Y. 10036
(A Non-Profit Organization)	

8. KNOWN BONDHOLDERS, MORTGAGEES, AND OTHER SECURITY HOLDERS OWNING OR HOLDING 1 PERCENT OR MORE OF TOTAL AMOUNT OF BONDS, MORTGAGES OR OTHER SECURITIES *(If there are none, so state)*

NAME	ADDRESS
None	

9. FOR OPTIONAL COMPLETION BY PUBLISHERS MAILING AT THE REGULAR RATES (Section 132.121, Postal Service Manual)

39 U. S. C. 3626 provides in pertinent part: "No person who would have been entitled to mail matter under former section 4359 of this title shall mail such matter at the rates provided under this subsection unless he files annually with the Postal Service a written request for permission to mail matter at such rates."

In accordance with the provisions of this statute, I hereby request permission to mail the publication named in Item 1 at the reduced postage rates presently authorized by 39 U. S. C. 3626.

(Signature and title of editor, publisher, business manager, or owner) Vincent N. Trimarco Jr.
Director of Administrative Services

10. FOR COMPLETION BY NONPROFIT ORGANIZATIONS AUTHORIZED TO MAIL AT SPECIAL RATES *(Section 132.122, Postal Manual)*

(Check one)

The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes

☒ Have not changed during preceding 12 months

☐ Have changed during preceding 12 months

(If changed, publisher must submit explanation of change with this statement.)

11. EXTENT AND NATURE OF CIRCULATION	AVERAGE NO. COPIES EACH ISSUE DURING PRECEDING 12 MONTHS	ACTUAL NUMBER OF COPIES OF SINGLE ISSUE PUBLISHED NEAREST TO FILING DATE
A. TOTAL NO. COPIES PRINTED *(Net Press Run)*	16,140	15,250
B. PAID CIRCULATION 1. SALES THROUGH DEALERS AND CARRIERS, STREET VENDORS AND COUNTER SALES	0	0
2. MAIL SUBSCRIPTIONS	14,466	14,082
C. TOTAL PAID CIRCULATION	14,466	14,082
D. FREE DISTRIBUTION BY MAIL, CARRIER OR OTHER MEANS 1. SAMPLES, COMPLIMENTARY, AND OTHER FREE COPIES	51	51
2. COPIES DISTRIBUTED TO NEWS AGENTS, BUT NOT SOLD	0	0
E. TOTAL DISTRIBUTION *(Sum of C and D)*	14,517	14,133
F. OFFICE USE, LEFT-OVER, UNACCOUNTED, SPOILED AFTER PRINTING	1,623	1,117
G. TOTAL *(Sum of E & F—should equal net press run shown in A)*	16,140	15,250

I certify that the statements made by me above are correct and complete.

(Signature of editor, publisher, business manager, or owner)
Vincent N. Trimarco Jr.
Director of Administrative Services

Contributions to the Journal of the Association for Computing Machinery

The *Journal of the Association for Computing Machinery* is a publication medium for original research papers of lasting value in the computer field. Submissions, which should be relevant to the interests of the Association, are judged primarily on originality and relevance. Contributions should conform to generally accepted practices for scientific papers with respect to organization and style of writing.

Papers may be sent to any area editor or to the editor-in-chief. Unless otherwise specified, they will be considered only for publication in the *Journal.* Until appearance or until final action, authors are expected to keep the editor informed of any changes of address.

Format. Manuscripts should be submitted in triplicate (the original on bond-weight paper) under cover of a submittal letter signed by the author. The text should be double spaced *on one side* of the paper. Typed manuscripts are preferred, but good reproductions of internal reports are acceptable (if text runs on both sides of pages, submit four copies). Authors' names should be given without titles or degrees. The name and address of the organization for which the work was carried out should be given. If the paper has previously been presented at a technical meeting, this fact, giving the date and sponsoring society, should appear in a footnote on the first page. Acknowledgments of funding sources should also be given in a footnote on the first page.

The usefulness of articles published in ACM periodicals is greatly enhanced when each paper includes information which insures proper indexing, classification, retrieval, and dissemination. To this effect authors should include in the manuscript:

(a) descriptive title;
(b) author names—with addresses in a footnote;
(c) informative abstract;
(d) content indicators of two types:
  (i) appropriate key words and key phrases,
  (ii) category numbers from *Computing Reviews (CR);*
(e) citations to the relevant literature.

The following suggestions may be useful in preparing this information.

Descriptive Title. Use a specific and informative title to tell accurately and clearly what the document is about. Choose title terms as highly specific as content and emphasis of the paper permit. Typically, a title might contain six to twelve words. Avoid special symbols and formulas in titles unless essential to indicate content. "Cute" or "clever" titles are unhelpful and should not be used.

Informative Abstract. The abstract should consist of short, direct, and complete sentences. A reading of the abstract should serve in some cases as a substitute for reading the paper itself. For this reason, the abstract should be *informative*. Typically, its length might be 150-200 words. The abstract should state the objectives of the work, summarize the results, and give the principal conclusions and recommendations. It should state clearly whether the focus is on theoretical developments or on practical questions, and whether subject matter or method are emphasized. The title need not be repeated. Work planned but not done should *not* be described in the abstract. Because abstracts are extracted from a paper and used separately, one should not use the first person, not display mathematics, and not use citation reference numbers. Try to avoid starting with the words "This paper . . ."

Content Indicators. Two types of content indicators are to be assigned: category numbers from the classification schedule used by *Computing Reviews,* and free choice key words and key phrases consisting of English language words. The latest *CR* classification may be found in any current *CR* issue, or in the *Journal of the ACM,* July 1973 issue.

Use as many category numbers as may be applicable. If possible, specify your interpretation of the "miscellaneous" or "general" categories if these are used. The following category numbers might, for example, be applicable to a manuscript dealing with sorting techniques: 3.74 (searching), 4.49 (miscellaneous utility programs), 5.31 (sorting).

In listing *key words* and *key phrases* to be used for indexing, put yourself in the place of the person who is looking for information in your index. If you have a technical thesaurus available, such as the *IFIP-ICC Vocabulary of Information Processing* [North-Holland Publishing Co., Amsterdam], consult it. Also, for helpful suggestions for alternate key words consult the citations to the relevant literature. The key words and key phrases used should be as precise as possible and hopefully unambiguous in their particular context. Typically ten to fifteen words or phrases might be used. The following additional guidelines may be of help:

(a) use important terms from the title; include also their synonyms, related words, and words of higher or lower generic rank;

(b) use English nouns, or noun-noun and noun-adjective combinations; do *not* use prepositions; do *not* use sequences of more than three words; do *not* use hyphens except if the hyphenated parts are *always* treated as a single unit;

(c) use specific terms whose meaning is generally accepted in the computer field; do *not* use broad catchall terms (such as "computer," "automatic," "machine," "system," "discussion," "description"); do not use private terms or acronyms that may not be generally known;

(d) do *not* use negative terms stressing what your paper does not do; emphasize the positive content and contribution.

Citations. (1) *References to items in periodicals:* These should take the form: author, title, journal, volume number, date, pages. For authors, last names are given first, even for multiple authors; likewise for editors, with the name followed by: (Ed.). The author's name always ends with a period, either the period which is the abbreviation for his initial, or a period for the purpose. The title has only the first word and proper names (or their derivatives) starting with capital letters, and it ends with a period. The date is given in parentheses. Example:

JONES, R.W., MARKS, F.W., AND ANTHONY, T. Programming routines for Boolean functions. *J. ACM 5* (May 1960), 5-19.

(2) *References to reports or proceedings:* Author(s) name(s) and title (same style as above), report number, source including date and pages.

(3) *References to books:* Author(s)—same style as to periodicals. Title—all principal words start with a capital letter, and the title is underlined so that it will be set in italics. Publisher, city, year. Page or chapter references follow the year.

(4) *In lengthy bibliographies,* entries must be arranged alphabetically according to authors' or editors' names, or publishing organizations for items to which no names can be attached.

Figures. Diagrams should be on smooth white paper or drafting linen. Lettering should be done professionally with a LeRoy ruler (or, if necessary, in clear, black typing). Photographs should be glossy prints. The author's name and the figure number should appear on the back of each figure. On publication, figures will be reduced to 4¾ inches in width; maximum allowable printed height will be 7½ inches. In planning, care should be taken to ensure that the legends and labels within the figure will be large enough to be readable after they are reduced by the same percentage as is required to make the whole figure fit on the page.

Mathematical Expressions. It will considerably lower the cost of composing type if you will:

(1) Avoid the use of buit up fractions; i.e. instead of $\frac{1}{n}$, use the negative exponent form n^{-1}; or instead of $\frac{1}{(1+n)^2}$, use $(1+n)^{-2}$; or instead of $\frac{1}{n}$, use 1/n. If not avoided by the author, built up fractions will be converted to equivalent expressions on the line when the paper is marked for the printer.

(2) Avoid the use of small-type mathematical expressions centered above or below arrows. If possible, try to use an alternative format.

(3) In the exponential function, avoid exponents having more than one or two characters; i.e. instead of $e^{x^2+y^2}$, use $\exp(x^2+y^2)$.

(4) Avoid the use of reference numbers for equations that are not subsequently referred to in the paper. The cost will be reduced if you will run the mathematical equations and other expressions in with the text (rather than displaying each on a separate line). In the *Journal,* the 5-inch wide type line makes it possible to accommodate long mathematical expressions on one line. Authors must expect that when accepted papers are marked for the printer, "excess" equation reference numbers will be deleted and equations will be run in with text.

(5) Show at the start what your underscored letter symbols indicate: italics, boldface (as in Algol), or typeset underscores; try to avoid the last as typeset underscores often require hand composition and opening up lines, and are thus expensive. In vector notations, indicate which letters or notations, if any, (i.e. $\overleftarrow{x}$) may be set in boldface type.

Copyright. Published articles are copyrighted by the Association for Computing Machinery, Inc. If material submitted for publication has been previously copyrighted, appropriate releases should accompany submitted papers; copyright notices will be inserted when reprinting such material.

Page Charge. Author's institutions or corporations are requested to honor a page charge of $35.00 per printed page or part thereof, to help defray the cost of publication. Charges are levied on all voluntarily contributed research papers, with 50 reprints of each paper furnished free of charge. Payment of page charges is not a condition of publication; editorial acceptance of a paper is unaffected by the payment or nonpayment.