CONSTRAINED STRING EDITING

B. John Oommen[+]

SCS-TR-48
May 1984

[+] School of Computer Science, Carleton University
Ottawa, Ontario K1S 5B6 Canada

A preliminary version of the paper will be presented at the 1984 Conference on the Mathematical Foundations of Computer Science, Prague, Czechoslovakia. Partially supported by the Natural Sciences and Engineering Research Council of Canada.

CONSTRAINED STRING EDITING[+]

B.J. OOMMEN[*]

Let X and Y be any two strings of finite length. We consider the problem of transforming X to Y using the edit operations of deletion, insertion and substitution. The optimal transformation is the one which has the minimum edit distance associated with it. The problem of computing this distance and the optimal transformation using no edit constraints has been studied in the literature. In this paper we consider the problem of transforming X to Y using any arbitrary edit constraints involving the number and type of edit operations to be performed. An algorithm has been presented to compute the minimum distance associated with editing X to Y subject to the specified constraint. The algorithim requires $O(|X|.|Y|.\min(|X|,|Y|))$ time and space. The technique to compute the optimal transformation has also been presented.

[+] Supported by the Natural Sciences and Engineering Research Council of Canada.

[*] School of Computer Science, Carleton University, Ottawa, K1S 5B6, Canada

I. INTRODUCTION

In the study of the automatic correction of garbled text a question that has interested researchers is that of quantifying the dissimilarity between two strings. A review of such distance measures and their application to string correction is given by Hall and Dowling [2] and Peterson [16].

The most promising of all the measures introduced seems to be the one that relates the strings using various edit operations. The edit operations usually considered are substitution, deletion, insertion and the reversal of adjacent letters. The studies of various researchers (see the literature surveys of references 2,6,9,10,16,18 and 19) indicate that most of the common errors that occur in garbled text consist of a single substitution, deletion, insertion or reversal error and hence the quantification of the dissimilarity between two strings using the above edit operations seems reasonable. Further, since a single reversal error can be modelled as a sequence of an insertion and a deletion error, the vast majority of research that has been conducted deals with merely the edit operations of substitution, deletion and insertion [2,5,6,7,8,9,10,15,16,17,19].

The distance referred to as the Generalized Levenshtein Distance (GLD) between two strings is defined as the minimum sum of the edit distances associated with the edit operations required to transform one string to the other. Apart from being a suitable index for comparing two strings this measure is closely related to the other numerical and non-numerical measures that involve the strings such as the Longest Common Subsequence, (LCS), and the Shortest Common Supersequence [7]. Various algorithms to compute this distance have been proposed the best of which are due to Wagner and Fischer [19] and Masek and Paterson [13]. For the infinite alphabet case it has been shown that Wagner and Fischer's algorithm is the optimal one [20]. A generalization of [19] has even been used to correct noisy substrings [8]. Closely related to these algorithms are the ones proposed to compute the Longest Common Subsequence (LCS) of two strings due to Hirschberg [3,4] Hunt and Szymanski [5] and Needleman and Wunsch [14]. Bounds on the complexity of the LCS problem have been given by Aho et al [1].

All of the above mentioned algorithms consider the editing of one string, say X, to transform it to Y with the edit process being absolutely unconstrained. The only algorithm known to us that considers constrained editing is the one due to Sankoff [17]. The latter algorithm is an LCS algorithm and involves a simple constraint that has its application in the comparison of Amino acid sequences.

In this paper we consider the problem of editing X to Y subject to far more general edit constraints. These edit constraints can be arbitrarily complex so long as they specified in terms of the number and type of the edit operations to be included in the optimal edit transformation. For the sake of clarity we give below some examples of constrained editing.

EXAMPLE 1:

Here are some typical constrained editing problems.

(a) What is the optimal way of editing X to Y using no more than k insertions?

(b) How can we optimally transform X to Y using exactly k substitutions?

(c) Is it possible to transform X to Y using exactly k_s substitutions, k_i insertions and k_e deletions. If it is possible, what is the distance between X and Y subject to this constraint?

In this paper we intend to propose an algorithm to compute the edit distance subject to the specified constraint. This is obtained by evaluating elements of a three-dimensional array W(.,.,.). The elements of W(.,.,.) are computed using dynamic programming techniques. After computing the array W(.,.,.) and edit distance subject to the constraint, the optimal sequence of edit operations can be obtained by backtracking through W(.,.,.). It will be shown that the algorithm requires $O(|X|.|Y|.\min(|X|,|Y|))$ time and space.

I.1. Notation

Let A be any finite alphabet and A* be the set of strings over A. θ is the null symbol, $\theta \notin A$. Let $\tilde{A} = A \cup \{\theta\}$. $\tilde{A}$ is referred to as the Appended Alphabet. A string X $\in$ A* of the form $X = x_1 \ldots x_N$, where each $x_i \in A$, is said to be of length $|X| = N$. Its prefix of length i will be written as X_i, $i \leq N$. Upper case symbols represent strings and lower case symbols, elements of the alphabet under consideration.

Let Z' be any element in $\tilde{A}^*$, the set of strings over $\tilde{A}$. The Compression Operator, C, is a mapping from $\tilde{A}^*$ to $\tilde{A}^*$. C(Z') is Z' withall occurrences of the symbol θ removed from Z'. Note that C preserves the order of the non-θ symbols in Z'. For example, if Z' = fθoθr, C(Z') = for.

I.2 The Set of Elementary Edit Distances d(.,.) and the Set $G_{X,Y}$

d(.,.) is a function whose arguments are a pair of symbols belonging to $\tilde{A}$, the appended alphabet, and whose range is the set of positive real numbers. d(θ, θ) is undefined and is not needed. The elementary distance d(a,b) can be interpreted as the distance associated with transforming 'a' to 'b', for a, b $\in \tilde{A}$. Thus,

(i) $d(x_i, y_j)$ is the distance associated with substituting x_i, by y_j, $x_i, y_j \in A$.

(ii) $d(x_i, \theta)$ is the distance associated with deleting $x_i \in A$.

(iii) $d(\theta, y_j)$ is the distance associated with inserting $y_j \in A$.

For every pair (X,Y), $X,Y \in \tilde{A}^*$, the finite set $G_{X,Y}$ is defined by means of the compression operator C, as a subset of $\tilde{A}^* \times \tilde{A}^*$.

$$G_{X,Y} = \{(X',Y') \mid (X',Y') \in \tilde{A}^* \times \tilde{A}^*, \text{ and obeys (i) - (iii)}\} \tag{1}$$

(i) $C(X') = X$, $C(Y') = Y$

(ii) $|X'| = |Y'|$

(iii) In no (X',Y') is $x'_i = y'_i = \theta$, $1 \leq i \leq |X'|$.

By definition, if $(X',Y') \in G_{X,Y}$, $\text{Max}[|X|,|Y|] \leq |X'| = |Y'| \leq |X| + |Y|$.

The meaning of the pair $(X',Y') \in G_{X,Y}$ is that it corresponds to one way of editing X into Y, using the edit operations of substitution, deletion and insertion. The edit operations themselves are specified for all $i = 1, \ldots, |X'|$ by (x'_i, y'_i), which represents the transformation of x'_i to y'_i. The cases below consider the three edit operations individually.

(i) If $x'_i \in A$ and $y'_i \in A$, it represents the substitution of x'_i by y'_i.

(ii) If $x'_i \in A$ and $y'_i = \theta$, it represents the deletion of x'_i. Between these two cases, all the symbols in X are accounted for.

(iii) If $x'_i = \theta$ and $y'_i \in A$, it represents the insertion of y'_i. Between cases (i) and (iii) all the symbols in Y are accounted for.

$G_{X,Y}$ is an exhaustive enumeration of the set of all the ways by which X can be edited to Y using the edit operations of substitution, insertion and deletion without destroying the order of the occurrence of the symbols in X and Y.

The number of elements in the set $G_{X,Y}$ is given by

$$|G_{X,Y}| = \sum_{k=\max[0,|Y|-|X|]}^{|Y|} \frac{(|X| + k)!}{k!(|Y|-k)!(|X|-|Y|+k)!}$$

Note that $|G_{X,Y}|$ depends only on $|X|$ and $|Y|$, and not on the actual strings X and Y themselves.

<u>EXAMPLE II.</u>

Let X = f and Y = go. Then,

$$G_{X,Y} = \{(f\theta,\ go),\ (\theta f,\ go),\ (f\theta\theta,\ \theta go),\ (\theta f\theta,\ g\theta o),\ (\theta\theta f,\ go\theta)\}$$

In particular the pair $(\theta f, go)$ represents the edit operations of inserting the 'g' and replacing the 'f' by an 'o'.

Since the Generalized Levenshtein Distance (GLD) between X and Y is the minimum of the sum of the edit distances associated with the edit operations required to transform X to Y, this distance, written as D(X,Y), has the expression [6,9],

$$D(X,Y) = \min_{(X',Y') \in G_{X,Y}} \left[\sum_{i=1}^{|X'|} d(x'_i, y'_i) \right] \qquad (2)$$

II. <u>EDIT</u> <u>CONSTRAINTS</u>

II.1 <u>Permissible and Feasible Edit Operations</u>

Consider the problem of editing X to Y, where $|X| = N$ and $|Y| = M$. Suppose we edit a prefix of X into a prefix of Y, using exactly i insertions, e deletions (or erasures) and s substitutions. Since the <u>number</u> of edit operations are specified, this corresponds to editing $X_{e+s} = x_1 \ldots x_{e+s}$, the prefix of X of length e+s, into $Y_{i+s} = y_1 \ldots y_{i+s}$, the prefix of Y of length i+s.

To obtain bounds on the magnitudes of the variables i, e and s, we observe that they are constrained by the lengths of the strings X and Y. Thus, if $r=e+s$, $q=i+s$ and $R=\text{Min}[M,N]$, these variables will have to obey the following obvious constraints.

$$\text{Max}[0,M-N] \le i \le q \le M$$

$$0 \le e \le r \le N$$

$$0 \le s \le \text{Min}[M,N]$$

Values of triples (i,e,s) which satisfy these constraints are termed as the <u>feasible values</u> of the variables. Let,

$$\left.\begin{aligned} H_i &= \{j \mid \mathrm{Max}[0,M-N] < j < M\}, \\ H_e &= \{j \mid 0 < j < N\}, \text{ and} \\ H_s &= \{j \mid 0 < j < \mathrm{Min}[M,N]\} \end{aligned}\right\} \qquad (3)$$

H_i, H_e and H_s are called the set of <u>permissible</u> values of i,e and s. Observe that a triple (i,e,s) is feasible if apart from $i \in H_i$, $e \in H_e$, $s \in H_s$, the following is satisfied:

$$i + s \leq M \quad \text{and} \quad e + s \leq N \qquad (4)$$

II.2 Specification of Edit Constraints

An edit constraint is specified in terms of the number and type of edit operations that are required in the process of transforming X to Y. It is expressed by formulating the number and type of edit operations in terms of three sets Q_e, Q_i and Q_s which are subsets of the sets H_i, H_e and H_s defined in (3). To clarify this, we consider the three constraints given in Example I.

<u>EXAMPLE III.</u>

(a) To edit X to Y performing no more than k insertions the sets Q_s and Q_e are both equal to $\emptyset$, the null set. Further,

$$Q_i = \{j \mid j \in H_i,\ j \leq k\}$$

(b) To edit X to Y performing exactly k substitutions Q_i and Q_e would be null and,

$$Q_s = \{k | k \in H_s\}$$

(c) To edit X to Y performing exactly k_i insertions, k_e deletions and k_s substitutions yields

$$Q_i = \{k_i | k_i \in H_i\},$$
$$Q_e = \{k_e | k_e \in H_e\}, \text{ and}$$
$$Q_s = \{k_s | k_s \in H_s\}.$$

<u>THEOREM</u> <u>I</u>.

Every edit constraint specified for the process of editing X to Y can be written merely as a subset of H_i.

<u>PROOF</u>. Let the edit constraint specified by the sets Q_i, Q_e and Q_s.

Every element $j \in Q_e$ requires the editing to be performed using exactly j deletions. Since $|X| = N$ this requires that the number of substitutions is N-j. Further, since $|Y| = M$, the number of insertions is forced to be M-N+j.

Similarly, if $j \in Q_s$, the edit transformations must contain exactly j substitutions. Since $|Y| = M$ and $|X| = N$, this requires that M-j insertions and N-j deletions are performed.

Let $\overline{Q}_e = \{M-N+j | j \in Q_e\}$, and

$\overline{Q}_s = \{M-j | j \in Q_s\}$

Thus, for any arbitrary constraint the number of insertions that are permitted is given by a set of integers which is obtained by performing unions and intersections on Q_i, $\overline{Q}_e$ and $\overline{Q}_s$, which is obviously a subset of H_i.

***

REMARK 1. The set referred to above which describes the constraint and which is the subset of H_i shall in future be written as T.

2. The edit constraint can just as easily be written as a subset of H_s or H_e. We have however chosen to describe T as a subset of H_i. This choice is absolutely subjective.

EXAMPLE IV.

Let X = for and Y = ga. Suppose we want to transform X to Y by performing at least 1 insertions, at most 1 substitutions and exactly 2 deletions. Then,

$Q_i = \{1,2\}$, $\quad Q_e = \{2\}$ and $\quad Q_s = \{0,1\}$

Hence,

$\overline{Q}_e = \{1\}$ and $\overline{Q}_s = \{1,2\}$

Thus $T = Q_i \wedge \overline{Q}_e \wedge \overline{Q}_s = \{1\}$

Hence the optimal transformations must contain <u>exactly</u> one insertion. Some candidate edit transformations are given by the following subset of $G_{X,Y}$.

$$\{(\theta for, g\theta\theta a), (f\theta or, ga\theta\theta), (fo\theta r, g\theta a\theta), (for\theta, g\theta\theta a)\}$$

Note that every pair in the above subset corresponds to at least one insertion, at most 1 substitution and exactly 2 deletions.

***

We shall refer to the edit distance subject to the constraint T as $D_T(X,Y)$. By definition, $D_T(X,Y) = \infty$ if $T = \emptyset$. This is merely a simple way of expressing that it is impossible to edit X to Y subject to the constraint T. We shall now consider the computations of $D_T(X,Y)$.

II. <u>W: THE ARRAY OF CONSTRAINED EDIT DISTANCES</u>

Let W (i,e,s) be the constrained edit distance associated with editing X_{e+s} to Y_{i+s} subject to the constraint that exactly i insertions, e deletions and s substitutions are performed in the process of editing. Let r=e+s and q=i+s. Let $G_{i,e,s}(X,Y)$ be the subset of the pairs in G_{X_r, Y_q} in which every pair corresponds to i insertions, e deletions and s substitutions. Since we shall consistently be referring to the strings X and Y, we refer to this set as $G_{i,e,s}$. Thus, using the notation of (1), W(i,e,s) has the expression,

$$W(i,e,s) = \min_{(X'_r, Y'_q) \in G_{i,e,s}} \left[\sum_{j=1}^{|X'_r|} d(x'_{rj}, y'_{qj}) \right] \text{ if } i,e \text{ or } s>0 \qquad (5)$$

We shall prove that the array W(.,.,.) is recursively computable.

THEOREM II.

Let W(i,e,s) be the quantity defined as in (5) for any two strings X and Y. Then,

$$W(i,e,s) = \text{Min}[\{W(i-1,e,s) + d(\theta,y_{i+s})\},\{W(i,e-1,s) + d(x_{e+s},\theta\},$$
$$\{W(i,e,s-1) + d(x_{e+s}, y_{i+s})\}]$$

for all feasible triples (i,e,s).

***

The theorem is proved in the Appendix.

The computation of the distance $D_T(X,Y)$ from the array W(i,e,s) only involves combining the appropriate elements of the array using T, the set of the number of insertions permitted. This is proved in the following theorem.

THEOREM III.

The quantity $D_T(X,Y)$ is related to the elements of the array W(i,e,s) as below:

$$D_T(X,Y) = \min_{i \in T} [(W(i,N-M+i,M-i)]$$

PROOF. Consider the constraints imposed on feasible values of i,e, and s. Since we are interested in the editing of the entire string X to the entire string Y, we have to consider only those elements of W(i,e,s) in which i+s=N and e+s=M. But the number of insertions can take any value from Max[0,M-N] to M. For every value of i in this range, the term in W(i,e,s) that can give a contribution to $D_T(X,Y)$ must have exactly M-i substitutions. Since the sum of the number of substitutions and the number of deletions is N, the term in W(i,e,s) that can give a contribution to $D_T(X,Y)$ must have exactly N-M+i deletions. Hence we are only interested in terms of the form W(i,N-M+i,M-i). $D_T(X,Y)$ is the minimum of all the terms which can yield a contribution, and hence the result follows.

***

REMARK. In an unconstrained editing problem i can take on any value ranging from Max[0,N-M] to M. Thus the unconstrained edit distance (which is commonly known as the Generalized Levenshtein Distance) D(X,Y), has the expression:

$$D(X,Y) = \min_{i \in [\mathrm{Max}[0,M-N],M]} [W(i,N-M+i,M-i)]$$

Using the results of the above two theorems, we now propose a computational scheme for $D_T(X,Y)$.

III. THE COMPUTATION OF W(.,.,.) AND $D_T(X,Y)$

To compute $D_T(X,Y)$ we make use of the fact that though the latter index itself does not seem to have any recursive properties, the index W(.,.,.), which is closely related to it has the interesting properties proved in Theorem II. Algorithm I, which we now propose, computes the array W(.,.,.) for all feasible values of the variables i,e and s.

Subsequently, using the array W(i,e,s) as the input, Algorithm II computes $D_T(X,Y)$ by comparing the contributions of the pertinent elements in W(.,.,.) as specified by Theorem III.

The computation of the array W(.,.,.) has to be done in a systematic manner, so that any quantity W(i,e,s) is computed before its value is required in any further computation. This is easily done by considering a three-dimensional coordinate system whose axes are i, e and s respectively. Initially the weight associated with the origin W(0,0,0) is assigned the value zero and the weights associated with the vertices on the axes are evaluated. Thus, W(i,0,0), W(0,e,0) and W(0,0,s) are computed for all permissible values of i,e, and s. Subsequently, the i-e, e-s and i-s planes are traversed, and the weights associated with the vertices on these planes are computed using the previously computed values. Finally, the weights corresponding to strictly positive values of the variables are computed. To avoid unnecessary computations, at each stage, the variables are tested for feasibility using the constraints of (4). The quantity $D_T(X,Y)$ is evaluated by comparing the weights associated with the points that lie on the three-dimensional line given by the parametric equation:

$$i = i; \quad e = N-M+i; \quad s = M-i$$

The algorithm to compute W(.,.,.) is given below.

<u>ALGORITHM</u> <u>I</u>

<u>Input</u>: The strings $X = x_1x_2\ldots x_N$, $Y = y_1y_2\ldots y_M$, and the set of elementary edit distances defined by d(.,.). Let R=Min[M,N].

<u>Output</u>: The array W(i,e,s) for all feasible values of i, e and s.

<u>Method</u>:

```
W(0,0,0)=0
for i=1 to M do   $W(i,0,0) = W(i-1,0,0) + d(\theta,y_i)$
for e=1 to N do   $W(0,e,0) = W(0,e-1,0) + d(x_e,\theta)$
for s=1 to R do   $W(0,0,s) = W(0,0,s-1) + d(x_s,y_s)$
for i=1 to M do
   for e=1 to N do
      $W(i,e,0) = \mathrm{Min}[W(i-1,e,0)+d(\theta,y_i), W(i,e-1,0)+d(x_e,\theta)]$
   end
end
```

```
for i=1 to M do
    for s=1 to M-i do
        $W(i,0,s) = Min[W(i-1,0,s)+d(\theta,y_{i+s}),W(i,0,s-1)+d(x_s,y_{i+s})]$
    end
end
for e=1 to N do
    for s=1 to N-e do
        $W(0,e,s) = Min[W(0,e-1,s)+d(x_{s+e},\theta),W(0,e,s-1)+d(x_{s+e},y_s)]$
    end
end
for i=1 to M do
    for e=1 to N do
        for s=1 to Min[(M-i), (N-e)] do
    $W(i,e,s) = Min[\{W(i-1,e,s)+d(\theta,y_{i+s})\},\{W(i,e-1,s)+d(x_{e+s},\theta)\},$
                    $\{W(i,e,s-1)+d(x_{e+s},y_{i+s})\}]$
        end
    end
end
END Algorithm
```

We now present Algorithm II which has for its input the array W(.,.,.), and yields as its output the quantity $D_T(X,Y)$.

ALGORITHM II.

Input: The array W(.,.,.) computed using Algorithm I, and the constraint set, T.

Output: The constrained distance $D_T(X,Y)$.

Method:

```
$D_T(X,Y) = \infty$
for all $i \in T$ do
    $D_T(X,Y) = \mathrm{Min}[D_T(X,Y),\ W(i,N-M+i,M-i)]$
end
END Algorithm II.
```

REMARKS: 1. The computational complexity of algorithms involving the comparison of two strings is conveniently given by the number of symbol comparisons required by the algorithm, [1,9,20]. In this case, the number of symbol comparisons required by Algorithm I has an upper bound of #(Alg I) given below.

$$\#(\text{Alg I}) = \frac{(N-M)M(M+1)}{2} + \frac{M(M+1)(2M+1)}{6}, \qquad \text{if } N > M$$

$$= \frac{(M-N)N(N+1)}{2} + \frac{N(N+1)(2N+1)}{6}, \qquad \text{otherwise}$$

Note that for every symbol comparison, we will need at most three multiplications and two additions. From the last set of for-loops it is easy to see that the time required is to compute the array W(.,.,.) is of O(MNR), where R=Min[M,N]. Thus it has a worst case complexity which is cubic in time. Algorithm II clearly requires linear time. Thus the overall time required to compute $D_T(X,Y)$ is O(MNR).

2. The array $W(i,e,s)$ contains far more information than is required to merely compute $D_T(X,Y)$. It contains all the weights associated with editing prefixes of X into prefixes of Y. One could therefore use the contents of the array to compute far more complicated indices which concern constrained edit distances involving prefixes of either or both the strings. We illustrate the computation of $D_T(X,Y)$ with an example.

EXAMPLE V.

Let X=aa and Y=bc. Let us suppose we want to edit X to Y permitting either 2 substitutions or exactly one edit operation of each type. It can be seen that the set T, defining the constraint is {0,1}.

We shall now follow through the computation of $D_T(aa,bc)$ using Algorithms I and II. To begin with, the weight associated with the origin is initialized. The i,e and s axes are then traversed.

$W(1,0,0) = d(\theta,b)$ $\quad$ $W(2,0,0) = d(\theta,b)+d(\theta,c)$

$W(0,1,0) = d(a,\theta)$ $\quad$ $W(0,2,0) = 2d(a,\theta)$

$W(0,0,1) = d(a,b)$ $\quad$ $W(0,0,2) = d(a,b)+d(a,c)$

The i-e, i-s and e-s planes are then traversed.

$W(1,1,0) = d(\theta,b)+d(a,\theta)$ $\quad$ $W(1,2,0) = d(\theta,b)+2d(a,\theta)$

$W(2,1,0) = d(\theta,b)+d(\theta,c)+d(a,\theta)$

$W(2,2,0) = d(\theta,b)+d(\theta,c)+2d(a,\theta)$

$W(1,0,1) = \text{Min}[d(\theta,b)+d(a,c),\ d(\theta,c)+d(a,b)]$

$W(0,1,1) = \text{Min}[d(a,\theta)+d(a,b),\ d(a,\theta)+d(a,c)]$

Finally, the weight for strictly positive values of i,e and s are computed.

$$W(1,1,1) = \text{Min}[\{d(\theta,b)+d(a,\theta)+d(a,c)\},\{d(\theta,c),d(a,\theta),d(a,b)\}]$$

Since the terms that must be compared to obtain $D_T(X,Y)$ are those which involve i=0 and i=1, $D_T(X,Y)$ can now be trivially computed as the minimum of W(0,0,2) and W(1,1,1). ***

III.1 Computing the Best Edit Sequence

Once the quantity $D_T(X,Y)$ has been computed the optimal edit sequence can be obtained by backtracking through the array W(.,.,.) and printing out the actual edit sequence traversed, in the reverse order. The technique is well known in dynamic programming problems and has been used extensively for edit sequences [13,15,19] and longest common subsequences [3,4,5,6]. Without further comment we now present Alogorithm III which has as its input the distance $D_T(X,Y)$ and the optimal element of W(.,.,.), i.e., the element W(I,E,S) which is equal to $D_T(X,Y)$.

ALGORITHM III

Input: The indices I,E and S for which $D_T(X,Y) = W(I,E,S)$

Output: The optimal sequence of edit operations subject to the specified constrained. The sequence is given in the reverse order and in the following notation.

a) The pair (x_i, y_j) means the substitution of x_i by y_j.

b) The pair (x_i, θ) means the deletion of x_i.

c) The pair (θ, y_j) means the insertion of y_j.

Method:

```
    $i = I$;    $e = E$;    $s = S$  where $W(I,E,S) = D_T(X,Y)$
    while ($i \neq 0$ and $e \neq 0$ and $s \neq 0$) do
    begin
      if ($W(i,e,s) = W(i-1,e,s) + d(\theta,y_{i+s})$) then
      begin
            print ($\theta,y_{i+s}$)
            $i = i-1$
      end
      else if ($W(i,e,s) = W(i,e-1,s) + d(x_{e+s},\theta)$) then
            begin
                print ($x_{e+s},\theta$)
                $e = e-1$
            end
            else
            begin
                print ($x_{e+s},y_{i+s}$)
                $s = s-1$
            end

    end
    END Algorithm III.
```

Remark. Obviously Algorithm III is performed in $O(\mathrm{Max}(M,N))$ time.

Applications of Constrained String Editing

A word concerning the applications of the results of this paper is not out of place. Let H be a finite dictionary and let Y be the noisy version of some unknown word X^+ in H. It is required to estimate X^+ by processing Y and comparing Y with every element of H.

It has been shown in the literature [9,15] that the GLD can be used as a measure of dissimilarity to compare Y with the individual elements of H. Fairly resonable correction results have been obtained using the GLD to estimate X^+. However the GLD has one minor drawback and that is that it weights the individual edit operations equally. Thus, for example, the insertion of an 'a' is given the same weight, whether it is the second insertion or the thousandth insertion. This is not always intuitively appealing. A more satisfactory way of computing the edit distance in cases when there are gross differences between the lengths of X and Y, would be one of comparing the strings using the constrained edit distance. For example, a good estimate of X^+ would be the string $X \in H$ which minimized the edit distance subject to the constraint that the number of insertions is "close to" $|Y| - |X|$. It is intuitively plausible that a constrained edit distance would be a better measure of the dissimilarity between the strings being considered. The actual constraint to be used would be dependent on the properties of the garbling mechanism - i.e., the properties of the error generating mechanism which transforms X^+ to Y.

Using the concepts introduced here we can also conceive of constrained measures related to the LCS of two strings. These measures would impose more generalized constraints than those that have been studied in the literature [17] and could be used to study the similarity between biological molecules which have gross differences in the lengths of their string representations.

CONCLUSIONS

In this paper we have considered the problem of editing a string X to a string Y subject to a specified edit constraint. The edit constraint is fairly arbitrary and can be specified in terms of the number and type of edit operations desired in the optimal transformation. The way by which the constraint, T, can be specified has been proposed. Also the technique to compute $D_T(X,Y)$, the edit distance subject to the constraint T, has been presented. A final algorithm has been given which has as its input the quantity $D_T(X,Y)$ and outputs the optimal edit transformation subject to the specified constraint.

Given the strings X and Y, $D_T(X,Y)$ and the array of constrained edit distances, W(.,.,.), can be computed in $O(|X|.|Y|.\mathrm{Min}(|X|,|Y|))$ time. If $D_T(X,Y)$ is given the optimal edit transformation can be obtained by backtracking through W(.,.,.) in $O(\mathrm{Max}(|X|,|Y|))$ time.

REFERENCES

[1] Aho, A.V., Hirschberg, D.S., and Ullman, J.D., "Bounds on the Complexity of the Longest Common Subsequence Problem," J-ACM, VOl. 23, 1976, p. 1-12.

[2] Hall, P.A.V., and Dowling, G.R., "Approximate String Matching", Computing Surveys, Vol.12, 1980, pp. 381-402.

[3] Hirschberg, D.S., "Algorithms for the Longest Common Subsequence Problem," J-ACM, Vol. 24, 1977, p. 664-675.

[4] Hirschberg, D.S., "A Linear Space Algorithm for Computing Maximal Common Subsequences," C-ACM, Vol. 18, 1975, pp. 341-343.

[5] Hunt, J.W., and Szymanski, T.G., "A Fast Algorithm for Computing Longest Common Subsequences," C-ACM, Vol. 20, 1977 , pp. 350-353.

[6] Kashyap, R.L., and Oommen, B.J., "A Common Basis for Similarity and Dissimilarity Measures Involving Two Strings", The International Journal of Computer Mathematics, Vol. 13, March 1983, pp. 17-40.

[7] Kashyap, R.L., and Oommen, B.J., "Similarity Measures for Sets of Strings". The International Journal of Computer Mathematics, Vol. 13, May 1983, pp. 95-104.

[8] Kashyap, R.L., and Oommen, B.J., "The Noisy Substring Matching Problem", IEEE Trans. on Software Engg., Vol. SE-9, 1983, pp. 365-370.

[9] Kashyap, R.L., and Oommen, B.J., "An Effective Algorithm for String Correction using Generalized Edit Distances -I. Description of the Algorithm and its Optimatlity", Information Sciences, VOl. 23, No. 2, March 1981, pp. 123 - 142.

[10] Kashyap, R.L., and Oommen, B.J., "Probabilistic Correction of Strings", Proc. of the IEEE Trans. on Pat. Recog. and Image Processing, June 1982, pp. 28-33.

[11] Levenshtein, A., "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," Sov. Phy. Dokl., Vol. 10, 1966, pp. 707-710.

[12] Maier, D., "The Complexity of Some Problems on Subsequences and Supersequences," J-ACM, VOl. 25, 1978, pp. 322-336.

[13] Masek, W.J., and Paterson, M.S., "A Faster Algorithm Computing String Edit Distances," J. of Comp. and Syst. Sci., Vol 20., 1980, pp. 18-31.

[14] Needleman, S.B., and Wunsch, C.D., "A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins," J. Mol. Biol., 1970, pp. 443-453.

[15] Okuda, T., Tanaka, E., and Kasai, T., "A Method of Correction of Garbled Words Based on the Levenshtein Metric", IEEE Trans. Comp., C-25, 1976, pp. 172-177.

[16] Peterson, J.L., "Computer Programs for Detecting and Correcting Spelling Errors," C-ACM, Vol. 23, 1980, pp. 676-687.

[17] Sankoff, D., "Matching Sequences under Deletion/Insertion Constraints", Proc. of the Nat. Acad. Science., USA, Vol. 69, Jan. 1972, pp. 4-6.

[18] Sellers, P.H., "An Algorithm for the Distance between Two Finite Sequences," J. of Combinatorial Theory, Vol. 16, 1974, pp. 253-258.

[19] Wagner, R.A., and Fischer, M.J., "The String to String Correction Problem," J-ACM, Vol. 21, 1974, pp. 168-173.

[20] Wong, C.K., and Chandra, A.K., "Bounds for the String Editing Problem," J-ACM, Vol. 23, 1976, pp. 13-16.

APPENDIX

Proof of Theorem II

It is required to prove that for all feasible values of i, e and s,

$$W(i,e,s) = \mathrm{Min}[\{W(i-1,e,s)+d(\theta,y_{i+s})\},\{W(i,e-1,s)+d(x_{e+s},\theta)\},$$
$$\{W(i,e,s-1)+d(x_{e+s},y_{i+s})\}]$$

The proof of the theorem is divided into three main divisions, written as Cases (a) - (c) respectively.

Case (a): Any two of the three variables i, e and s are zero.
Case (b): Any one of the three variables i, e and s are zero.
Case (c): None of the variables i, e and s are zero.

The most involved of these cases is Case (c). Cases (a) and (b) are merely one and two parameter cases respectively of Case (c). To avoid repetition, we shall here prove only Case (c). Thus, for the rest of the proof, we encounter only strictly positive values for the variables i, e and s. Let $r=e+s$, $q=i+s$, $X_r=x_1 \ldots x_r$ and $Y_q=y_1 \ldots y_q$. By definition,

$$W(i,e,s) = \underset{(X'_r\ ,\ Y'_q)}{\mathrm{Min}} \sum_{j=1}^{|X'_r|} d(x'_{rj},\ y'_{qj}) \qquad (A.1)$$

where, (X'_r,Y'_q) is the arbitrary element of the set $G_{i,e,s}$, with x'_{rj} and y'_{qj} as the jth symbols of X'_r and Y'_q respectively. Let the lengths of the strings X'_r and Y'_q in the arbitrary element be L. Then the last symbols of X'_r and Y'_q are x'_{rL} and y'_{qL} respectively.

We partition the set $G_{i,e,s}$ into three <u>mutually</u> <u>exclusive</u> <u>and</u> <u>exhaustive</u> subsets.

$$G^1_{i,e,s} = \{(X'_r,Y'_q) \mid (X'_r,Y'_q) \in G_{i,e,s},\ x'_{rL} = x_r,\ y'_{qL} = y_q\}$$

$$G^2_{i,e,s} = \{(X'_r,Y'_q) \mid (X'_r,Y'_q) \in G_{i,e,s},\ x'_{rL} = x_r,\ y'_{qL} = \theta\}$$

$$G^3_{i,e,s} = \{(X'_r,Y'_q) \mid (X'_r,Y'_q) \in G_{i,e,s},\ x'_{rL} = \theta,\ y'_{qL} = y_q\}$$

By their definitions, we see that the three above sets are mutually exclusive. Further, since x'_{xL} and y'_{qL} cannot be θ simultaneously, every pair in $G_{i,e,s}$ must be in one of the above sets. Hence these three sets partition $G_{i,e,s}$. Rewriting (A.1) we obtain,

$$W(i,e,s) = \operatorname*{Min}_{k=1,2,3} \left[\operatorname*{Min}_{(X'_r,\ Y'_q) \in G^k_{i,e,s}} S' \right] \qquad (A.2)$$

where $$S' = \sum_{j=1}^{|X'_r|} d(x'_{rj},\ y'_{qj})$$

Consider each of the terms (A.2) individually. In every pair in $G^1_{i,e,s}$, $x'_{rI} = x_r$ and $y'_{qI} = y_q$. Hence,

$$\min_{(X'_r,Y'_q)\in G^1_{i,e,s}} \sum_{j=1}^{|X'_r|} d(x'_{rj},y'_{qj})$$

$$= \left[\min_{(X'_r,Y'_q)\in G^1_{i,e,s}} \sum_{j=1}^{|X'_r|-1} d(x'_{rj},y'_{qj})\right] + d(x_r,y_q) \tag{A.3}$$

For every element in $G^1_{i,e,s}$ there is a unique element in $G_{i,e,s-1}$ and vice versa. Hence, the first term in above expression is exactly W(i,e,s-1). Since r=e+s and q=i+s,

$$\min_{(X'_r,Y'_q)\in G^1_{i,e,s}} \sum_{j=1}^{|X'_r|} d(x'_{rj},y'_{qj})$$

$$= W(i,e,s-1) + d(x_{e+s},y_{i+s}) \tag{A.4}$$

Consider the second term in (A.2). In every pair in $G^2_{i,e,s}$, $x'_{rL} = x_r$ and $y'_{qL} = \theta$. Hence,

$$\min_{(X'_r,Y'_q)\in G^2_{i,e,s}} \sum_{j=1}^{|X'_r|} d(x'_{rj},y'_{qj})$$

$$= \left[\min_{(X'_r, Y'_q) \in G^2_{i,e,s}} \sum_{j=1}^{|X'_r|-1} d(x'_{rj}, y'_{qj}) \right] + d(x_r, \theta) \tag{A.5}$$

For every element in $G^2_{i,e,s}$ there is a unique element in $G_{i,e-1,s}$ and vice versa. Hence, the first term in above expression is exactly W(i,e-1,s). Thus,

$$\min_{(X'_r, Y'_q) \in G^2_{i,e,s},} \sum_{j=1}^{|X'_r|} d(x'_{rj}, y'_{qj})$$

$$= W(i,e-1,s) + d(x_{e+s}, \theta) \tag{A.6}$$

Consider the third term in (A.2). In every pair in $G^3_{i,e,s}$, $x'_{rL} = \theta$ and $y'_{qL} = y_q$. Hence,

$$\min_{(X'_r, Y'_q) \in G^3_{i,e,s},} \sum_{j=1}^{|X'_r|} d(x'_{rj}, y'_{qj})$$

$$= \left[\min_{(X'_r, Y'_q) \in G^3_{i,e,s}} \sum_{j=1}^{|X'_r|-1} d(x'_{rj}, y'_{qj}) \right] + d(\theta, y_q) \tag{A.7}$$

For every element in $G^3_{i,e,s}$ there is a unique element in $G_{i-1,e,s}$ and vice versa. Hence, the first term in above expression is exactly W(i-1,e,s). As in the above cases,

$$\min_{(X'_r,Y'_q)\in G^3_{i,e,s},} \sum_{j=1}^{|X'_r|} d(x'_{rj},y'_{qj})$$

$$= W(i-1,e,s) + d(\theta,y_{i+s})$$

Resubstituting (A.4), (A.6) and (A.8) in (A.2) proves the theorem.