Similarity Measures for Sets of Strings

R.L. Kashyap* and B.J. Oommen**

SCS-TR-15
January 1983

* Department of Electrical Engineering, Purdue University, W. Lafayette, IN, 47907, USA.

** School of Computer Science, Carleton University, Ottawa, Ontario, K1S 5B6, Canada.

To be published in the International Journal of Computer Mathematics, Vol. 13, No. 2, May 1983.

SIMILARITY MEASURES FOR SETS OF STRINGS[†]

Kashyap, R.L., and Oommen, B.J.,

ABSTRACT

In the companion paper[3], we have presented a common basis for many of the similarity and dissimilarity measures involving a pair of strings. In this paper, we extend the results to capture various numerical and nonnumerical measures involving more than two strings. A measure D(X,Y,...,Z) has been defined involving the set of strings {X,Y,...Z} in terms of two abstract operators $\oplus$ and $\otimes$ and a function $\delta(\cdot,\cdot)$ which has as many arguments as there are strings in the set {X,Y,...,Z}. The quantity D(X,Y,...,Z) represents various numerical and nonnumerical quantities involving {X,Y,...,Z} such as Length of their Longest Common Subsequence, (LLCS) the Length of their Shortest Common Supersequence, (LSCS) the set of their common subsequences, the set of their common supersequences and the set of their shuffles. The computational properties of D(X,Y,...,Z) have also been discussed.

I. Introduction

In the companion paper [3], we presented a common abstract foundation for many of the known similarity and dissimilarity measures that involve a pair of strings. Some of these measures include the LLCS of two strings, their LSCS and the set of Longest Common Subsequences (LCS). Many of the quantities defined for a pair of strings can easily be extended for a set of strings. In fact, all the quantities discussed in the companion paper [3]

[†] Partially supported by the National Science Foundation Grant No. ECS-80-09041.

Index Terms: Similarity measures for sets of strings, dissimilarity measures for sets of strings, common basis for properties involving sets of strings, subsequences, supersequences and shuffles of sets of strings.

with the exception of the Generalized Levenshtein Distance GLD(X,Y) and the probability P(Y/X), lend themselves to natural generalizations. For example, a subsequence common to X, Y and Z, of maximum length, would be one of their longest common subsequences. In this paper we consider the problem of defining and computing a family of similarity measures that involve a set of strings.

Since the theory of similarity measures for a pair of strings has been fairly well developed, one possible approach to studying more than two strings is to study them in a pairwise fashion. However, even simple numerical measures involving a set of strings cannot be obtained as a function of their corresponding pairwise measures. For example, even the LLCS of three strings X, Y, and Z, cannot be expressed in terms of three pairwise quantities, LLCS(X,Y), LLCS(Y,Z) and LLCS(X,Z). Specifically, if LLCS (X,Y), LLCS(Y,Z) and LLCS(X,Z) are known, the LLCS(X,Y,Z) is not determined. Similarly, if $X,Y,Z \in A^*$, it is easy to define the set of LCS of the three strings. Currently, there is no algorithm to compute this set. In fact, very little effort has gone in to efficiently compute or even to define similarity measures over a set of strings.

One of the applications of similarity measures involving sets of strings is in the clustering of records identified by their keywords [2]. Further, comparison of more than two molecular proteins can be achieved [1] by using any reasonable similarity measure involving them.

Considering all this, a unifying theory involving properties common to sets of strings is desirable. The abstract measure D(X,Y), introduced in the companion paper [3], has the advantage that it can be generalized for K strings {X,Y,....,Z}, yielding a measure involving them, D(X,Y,....,Z). As in the previous case, all the various measures between the K strings such as their LLCS, LSCS, the set of their common subsequences etc. can be seen to be particular cases of D(X,Y,...,Z). Further, since D(X,Y,...,Z) can be recursively computed all the above quantities can be computed using a common computational algorithm.

Among the new contributions of this paper are :

(1) An algorithm to compute numerical indices of similarity, such as the LLCS and LSCS, involving more than two strings.

(2) Algorithms to compute the set of all the LCS and the set of all the Shortest Common Supersequences (SCS) involving more than two strings. The algorithms presented in this case are also one-pass algorithms, that require no backtracking.

I.1. Notation

The notation and the terminology of the companion paper [3] is assumed. A is a finite alphabet, and λ is the null symbol, distinct from μ the null string. $\tilde{A} = A \cup \{\lambda\}$ is called the Appended Alphabet. $\oplus$ and $\otimes$ are two abstract operators, whose properties are discussed in [3]. Three structures that we will repeatedly use are the following [3] :

(a) $\tau_3 = (Z_p', \text{MAX}, +, 0, 0)$

(b) $\tau_4 = (T_A, \cup, \cdot, \emptyset, \{\mu\})$

(c) $\tau_5 = (Z_p', \text{MIN}, +, \infty, 0)$

where Z_p' is the set consisting of the nonnegative integers and ∞, T_A is the power set of A^*, and $\emptyset$ is the null set.

II. The Abstract System and the Abstract Measure for K Strings

Let β be a set of K strings, $\{X,Y,..,Z\}$. The abstract measure, D(X,Y,..,Z) involving the K strings is said to be induced by the system (A,τ,σ), explained below:

(i) τ is the algebraic structure defined in II.1 of [3].

(ii) δ is a map from $\tilde{A}^K$ to T, where T is the set defined in τ. δ has as many arguments as there are strings. The quantity $\delta(\lambda,\lambda,\ldots,\lambda)$ is undefined and is not needed.

The set $G_{X,Y,\ldots,Z}$ is defined as:

$$G_{X,Y,\ldots,Z} = \{(X',Y',\ldots,Z') \mid (X',Y',\ldots,Z') \in \tilde{A}^K, \text{and obeys (i)-(iii)}\}$$

(i) $C(X') = X,\ C(Y') = Y, \ldots,\ C(Z') = Z$

(ii) $|X'| = |Y'| = \ldots = |Z'|$

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

The set $G_{X,Y}$ defined in [3] for a pair of strings represents all the ways by which Y can be edited into X using only the edit operations of substitution, deletion and insertion. $G_{X,Y,\ldots,Z}$ defined above is a direct generalization of $G_{X,Y}$.

The measure $D(X,Y,\ldots,Z)$, induced by (A,τ,δ) is defined by (1).

$$D(X,Y,\ldots,Z) = I, \quad \text{the identity for } \otimes \text{, if } X = Y = \ldots = Z = \mu$$

$$= \bigoplus_{(X',Y',\ldots,Z') \in G_{X,Y,\ldots,Z}} \left[\bigotimes_{i=1}^{|X'|} d(x'_i, y'_i, \ldots, z'_i) \right], \quad \text{otherwise} \qquad (1)$$

The fact that $D(X,Y,\ldots,Z)$ is a unique element in T follows again directly from the associativity of $\otimes$ and the associativity and commutativity of $\oplus$. We shall now state some theorems regarding $D(X,Y,\ldots,Z)$ and observe some of the properties of the set $\{X,Y,\ldots,Z\}$ which it captures. These theorems are direct generalizations of the corresponding theorems for a pair of strings, and the reasoning behind their proofs follows the same reasoning as their earlier counterparts.

Theorem 1.

Let τ_3 be as defined in Section I.1, and let δ_1 be a function mapping $\tilde{A}^K$ to Z_p' as:

$$\delta_1(a_1,a_2,\ldots,a_K) = 1 \quad \text{if } a_1 = a_2 = \ldots a_K = a,\ a_i \in A$$
$$= 0 \quad \text{otherwise}$$
$$\delta_1(\lambda,\lambda,\ldots,\lambda) \text{ is undefined.}$$

Then the measure induced by the system (A,τ_3,δ_1) is exactly the LLCS of the set $\{X,Y,\ldots,Z\}$.

***

Theorem 2.

Let τ_4 be as defined as in Section I.1. Let δ_2 be a function mapping $\tilde{A}^K$ to the power set of A^* as:

$$\delta_2(a_1,a_2,\ldots,a_K) = \{a\} \quad \text{if } a_1 = a_2 = a_3 = \ldots a_K = a,\ a_i \in A$$
$$= \{\mu\} \quad \text{otherwise}$$
$$\delta_2(\lambda,\lambda,\ldots,\lambda) \text{ is undefined}$$

The measure induced by the system (A,τ_4,δ_2) is the <u>set</u> <u>of</u> <u>all</u> the subsequences common to <u>all</u> the strings $\{X,Y,\ldots,Z\}$.

***

Theorem 3.

Let τ_5 be as defined in Section I.1 and let δ_3 be a function mapping $\tilde{A}^K$ to Z_p' as:

$$\delta_3(a_1,a_2,\ldots,a_K) = \infty \quad \text{if any two non-}\lambda\text{ arguments are distinct.}$$

$=1$ if all non-Λ arguments are equal.

$\delta_3(\Lambda,\Lambda,\ldots,\Lambda)$ is undefined.

The measure induced by the system (A,τ_5,δ_3) is exactly the LSCS of the set $\{X,Y,\ldots Z\}$.

***

Theorem 4.

Let τ_4 be the structure defined in Section I.1. Let δ_4 be a function mapping $\tilde{A}^K$ to the power set of A^* as:

$$\delta_4(a_1,a_2,\ldots,a_K) = \emptyset \quad \text{if any two non-}\Lambda\text{ arguments are distinct.}$$
$$= \{a\} \quad \text{if all non-}\Lambda\ a_i\text{'s are equal to a .}$$

$\delta_4(\Lambda,\Lambda,\ldots,\Lambda)$ is undefined.

Then the measure induced by the system (A,τ_4,δ_4) is the set of all the supersequences common to all the K strings $\{X,Y,\ldots Z\}$.

***

Theorem 5.

Let τ_4 be as defined in Section I.1. Let δ_5 be a function defining the set of elementary measures, and which maps $\tilde{A}^K$ to the power set of A^* as:

$$\delta_5(a_1,a_2,\ldots,a_k) = \{a_i\} \quad \text{iff exactly one argument } a_i \text{ is non-}\Lambda$$
$$= \emptyset \quad \text{otherwise}$$

The measure induced by the system (A,τ_5,δ_5) is the set of all the shuffles of all the K strings $\{X,Y,\ldots,Z\}$.

III. Computation of the Abstract Measure for More Than Two Strings

The computation of the abstract measure for more than two strings follows arguments similar to the computation of the abstract measure for a pair of strings. The measure has analogous iterative properties, which are K-dimensional generalizations of the properties valid for a pair of strings. For the sake of simplicity, we shall consider the case when K = 3. We shall state a theorem regarding the measure D(X,Y,Z) in terms of the measures associated with their left derivatives, $\underline{X}$, $\underline{Y}$, and $\underline{Z}$ respectively, and their last symbols.

<u>Theorem 6</u>.

Let $X = x_1 \ldots x_N$, $Y = y_1 \ldots y_M$ and $Z = z_1 \ldots z_R$, $M,N,R \geq 1$. Then,

$D(X,Y,Z) =$

$$\left[D(\underline{X},Y,Z) \otimes \delta(x_N,\Lambda,\Lambda)\right] \oplus \left[D(X,\underline{Y},Z) \otimes \delta(\Lambda,y_M,\Lambda)\right] \oplus \left[D(X,Y,\underline{Z}) \otimes \delta(\Lambda,\Lambda,z_R)\right]$$

$$\oplus \left[D(\underline{X},\underline{Y},Z) \otimes \delta(x_N,y_M,\Lambda)\right] \oplus \left[D(\underline{X},Y,\underline{Z}) \otimes \delta(x_N,\Lambda,z_R)\right] \oplus \left[D(X,\underline{Y},\underline{Z}) \otimes \delta(\Lambda,y_M,z_R)\right]$$

$$\oplus \left[D(\underline{X},\underline{Y},\underline{Z}) \otimes \delta(x_N,y_M,z_R)\right]$$

***

The proof of the theorem is identical to that of Theorem 9 of [3]. By partitioning $G_{X,Y,Z}$ into 7 mutually exclusive and exhaustive sets, D(X,Y,Z) is split up as seven terms operated on by $\oplus$. After stripping off the last elements of the triples in $G_{X,Y,Z}$, these sets can be identified as $G_{\underline{X},Y,Z}$, $G_{X,\underline{Y},Z}$, $G_{X,Y,\underline{Z}}$, $G_{\underline{X},\underline{Y},Z}$, $G_{\underline{X},Y,\underline{Z}}$, $G_{X,\underline{Y},\underline{Z}}$ and $G_{\underline{X},\underline{Y},\underline{Z}}$. The theorem follows directly thereafter by using the associativity of $\otimes$ and the distributivity of $\otimes$ over $\oplus$. Using Theorem 6 and the 3-dimensional version of Theorem 9 of [3] we present Algorithm I to compute the measure D(X,Y,Z).

<u>Algorithm</u> <u>I</u>.

<u>Input</u>: Three strings $X = x_1 \ldots x_N$, $Y = y_1 \ldots y_M$, $Z = z_1 \ldots z_R$

<u>Output</u>: The Abstract measure $D(X,Y,Z)$ induced by the system (A,τ,δ)

<u>Method</u>:

```
$D(\mu,\mu,\mu) = I$
for i=1 to N do $D(X_i,\mu,\mu) = D(X_{i-1},\mu,\mu) \otimes \delta(x_i,\lambda,\lambda)$
for j=1 to M do $D(\mu,Y_j,\mu) = D(\mu,Y_{j-1},\mu) \otimes \delta(\lambda,y_j,\lambda)$
for k=1 to R do $D(\mu,\mu,Z_k) = D(\mu,\mu,Z_{k-1}) \otimes \delta(\lambda,\lambda,z_k)$
for i=1 to N do
    for j = 1 to M do
    $D(X_i,Y_j,\mu) = \left[D(X_{i-1},Y_j,\mu) \otimes \delta(x_i,\lambda,\lambda)\right] \oplus \left[D(X_i,Y_{j-1},\mu) \otimes \delta(\lambda,y_j,\lambda)\right]$
                  $\oplus \left[D(X_{i-1},Y_{j-1},\mu) \otimes \delta(x_i,y_j,\lambda)\right]$
    end
end
```

```
for i=1 to N do
    for k=1 to R do
    $D(X_i,\mu,Z_k) = \left[D(X_{i-1},\mu,Z_k) \otimes \delta(x_i,\lambda,\lambda)\right] \oplus \left[D(X_i,\mu,Z_{k-1}) \otimes \delta(\lambda,\lambda,z_k)\right]$
                $\oplus \left[D(X_{i-1},\mu,Z_{k-1}) \otimes \delta(x_i,\lambda,z_k)\right]$
    end
end
for j=1 1 to M do
    for k=1 to R do
    $D(\mu,Y_j,Z_k) = \left[D(\mu,Y_{j-1},Z_k)\otimes\delta(\lambda,y_j,\lambda)\right] \oplus \left[D(\mu,Y_j,Z_{k-1}) \otimes \delta(\lambda,\lambda,z_k)\right]$
                $\oplus \left[D(\mu,Y_{j-1},Z_{k-1}) \otimes \delta(\lambda,y_j,z_k)\right]$
    end
end
for i=1 to N do
    for j=1 to M do
        for k=1 1 to R do
$D(X_i,Y_j,Z_k) = \left[D(X_{i-1},Y_j,Z_k) \otimes \delta(x_i,\lambda,\lambda,)\right] \oplus \left[D(X_i,Y_{j-1},Z_k) \otimes \delta(\lambda,y_j,\lambda)\right]$
        $\oplus \left[D(X_i,Y_j,Z_{k-1}) \otimes \delta(\lambda,\lambda,z_k)\right] \oplus \left[D(X_{i-1},Y_{j-1},Z_k) \otimes \delta(x_i,y_j,\lambda)\right]$
        $\oplus \left[D(X_{i-1},Y_j,Z_{k-1}) \otimes \delta(x_i,\lambda,z_k)\right] \oplus \left[D(X_i,Y_{j-1},Z_{k-1}) \otimes \delta(\lambda,y_j,z_k)\right]$
        $\oplus \left[D(X_{i-1},Y_{j-1},Z_{k-1}) \otimes \delta(x_i,y_j,z_k)\right]$
        end
    end
end
return $D(X_N,Y_M,Z_R)$
END Algorithm I.
```

Just like Algorithm I of [3], this algorithm too can be understood in the context of a three dimensional graph--a three dimensional trellis. The nodes of the trellis are triples, which correspond to the prefixes of the strings. The edges of the trellis are directed, and have elementary measures associated with the transformations they represent. The quantity $D(X_i,Y_j,Z_k)$ is the measure associated with the node (X_i,Y_j,Z_k).

Initially the measures associated with the X, Y and Z axes are obtained, and then using the extended results of Theorem 9 of [3], the measures associated with the nodes on the X-Y, X-Z and Y-Z planes are computed. Finally the 3-dimensional trellis is traversed, the measure associated with a node being computed using the computed measures associated with the seven adjacent nodes from which it can be reached tracing at most one edge.

III.1. Remarks

(i) From the last set of for-loops, it is evident that Algorithm I requires $O(MNR)$ symbol comparisons. It performs the operations $\oplus$ and $\otimes$ $O(MNR)$ times, and thus has a time complexity of $O(MNR)$ if it is used to compute any of the numerical quantities discussed in this paper.

(ii) In particular cases (like the LCS problem) many of the individual elementary terms will have a magnitude of zero, and so for these particular cases, the algorithm can be simplified considerably.

(iii) Using arguments similar to those used in Theorem 6, an algorithm can be written to compute the measure between more than three strings. If the number of strings, K, is considered as a variable, it can be seen that the complexity of the resulting algorithm will be $O(2^K \prod_{i=1}^{K} N_i)$, where N_i is the length of the ith string. This follows from the fact the the algorithm will have to compute 2^K-1 terms in its last set of "for" loops. This agrees with the result of Maier [4], concerning the NP-completeness of the LCS and SCS problem for arbitrarily large alphabets. The question of whether this bound $O(2^K \prod_{i=1}^{K} N_i)$ can be reduced remains an open problem.

(iv) Consider the problem of computing the set of all LCS between X, Y, and Z. From Theorem 2, the measure induced between X, Y and Z by the system (A,τ_4,δ_2) is the set of all their common subsequences. By extracting from this set, the elements of longest length, the set of all LCS of the strings X, Y, Z can be obtained. Since the set of common subsequences can be iteratively computed, by pruning the individual measures between the prefixes of the strings and retaining only the elements of maximum length, the set of LCS can also be iteratively computed[†].

An alternate method of computing the set of LCS of X, Y and Z is by a pairwise comparison of the three strings. We have already shown [3], that $D^4(X,Y)$ and $D^4(Y,Z)$ are the sets of the common subsequences between X and Y, and Y and Z respectively. Hence, the intersection of the two sets, $D^4(X,Y)$ and $D^4(Y,Z)$ is the set of the subsequences common to X, Y and Z. From the latter set, the set of LCS of X, Y and Z can be trivially obtained. However, the three dimensional algorithm which processes X, Y and Z simultaneously is more efficient than the latter because, at every stage in the computation, it will retain only the LCS between the prefixes of all the three strings. Strings eliminated on intersecting $D^4(X,Y)$ and $D^4(Y,Z)$ will not even appear in the computation. We shall conclude this section with an example of the computation of the set of LCS between three strings.

Example 1. Let X = atoms, Y = tames and Z = atm, where A = {a,e,m,o,s,t}. The computation of the set of LCS of X, Y and Z is given in Fig. 1. The 3-dimensional trellis is represented by a sequence of arrays parallel to the X-Y plane. In the array corresponding to Z_k, the entry X_i,Y_j is the set of LCS between X_i, Y_j and Z_k. The details of the computation are omitted.

[†] As explained in Section V.2 of [3], the set of LCS of {X,Y,...,Z} can be obtained as a special case of D(X,Y,...,Z) by formalizing this extraction process using the operator Ⓤ described in [3].

	μ	a	t	o	m	s
μ	μ	μ	μ	μ	μ	μ
t	μ	μ	μ	μ	μ	μ
a	μ	a	a	a	a	a
m	μ	a	a	a	a	a
e	μ	a	a	a	a	a
s	μ	a	a	a	a	a

$Z_1 = a$

	μ	a	t	o	m	s
μ	μ	μ	μ	μ	μ	μ
t	μ	μ	t	t	t	t
a	μ	a	a,t	a,t	a,t	a,t
m	μ	a	a,t	a,t	a,t	a,t
e	μ	a	a,t	a,t	a,t	a,t
s	μ	a	a,t	a,t	a,t	a,t

$Z_2 = at$

	μ	a	t	o	m	s
μ	μ	μ	μ	μ	μ	μ
t	μ	μ	t	t	t	t
a	μ	a	a,t	a,t	a,t	a,t
m	μ	a	a,t	a,t	am,tm	am,tm
e	μ	a	a,t	a,t	am,tm	am,tm
s	μ	a	a,t	a,t	am,tm	am,tm

$Z_3 = atm$

Fig. 1 : The trellis for computing the LCS of X, Y and Z of Example 1. The required set is {am,tm} in the right bottom corner.

IV. Conclusions

In the companion paper [3] we presented a unifying theory for similarity properties involving a pair of strings, X and Y, and which required that the order of the symbols of both the strings are preserved. In this paper, we have extended these results, and given a common basis for many of the similarity measures involving more than two strings. We have also proposed a common computational scheme to compute many numerical and nonnumerical indices of similarity involving subsequences, supersequences and shuffles of a set of strings.

REFERENCES

1. Dayhoff, M. O., "Atlas of Protein Sequences and Structure", Vol. 5, Silver Springs, Maryland: National Biomedical Research Foundation, 1972.

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

3. Kashyap, R.L., and Oommen, B.J., "A Common Basis for Similarity Measures Involving Two Strings", To appear in Int. Journal of Comp. Math. Vol. 13.

4. Maier, D., "The Complexity of Some Problems on Subsequences and Supersequences," J-ACM, Vol. 25, 1978, pp. 322-336.

Carleton University, School of Computer Science
Bibliography of Technical Reports
Publications List

School of Computer Science
Carleton University
Ottawa, Ontario, Canada
KIS 5B6

SCS-TR-1	**The Design of CP-6 Pascal** Jim des Rivieres and Wilf R. LaLonde, June 1982
SCS-TR-2	**Single Production Elimination in LR(1) PARSERS: A Synthesis** Wilf R. LaLonde, June 1982
SCS-TR-3	**A Flexible Compiler Structure That Allows Dynamic Phase Ordering** Wilf R. LaLonde and Jim des Rivieres, June 1982
SCS-TR-4	**A Practical Longest Common Subsequence Algorithm for Text Collation** Jim des Rivieres, June 1982
SCS-TR-5	**A School Bus Routing and Scheduling Problem** Wolfgang Lindenberg, Frantisek Fiala, July 1982
SCS-TR-6	**Routing Without Routing Tables: Labelling and Implicit Routing in Networks** Nicola Santoro and Ramez Khatib, July 1982
SCS-TR-7 Out-of-print	**Concurrency Control in Large Computer Networks** Nicola Santoro and Jeffrey B. Sidney, July 1982
SCS-TR-8 Out-of-print	**Order Statistics on Distributed Sets** Nicola Santoro and Jeffrey B. Sidney, July 1982
SCS-TR-9	**Oligarchical Control of Distributed Processing Systems** Moshe Krieger and Nicola Santoro, August 1982
SCS-TR-10	**Communication Bounds for Selection in Distributed Sets** Nicola Santoro and Jeffrey B. Sidney, September 1982
SCS-TR-11	**A Simple Technique for Converting from a Pascal Shop to a C Shop** Wilf R. LaLonde and John R. Pugh, November 1982
SCS-TR-12	**Efficient Abstract Implementations for Relational Data Structures** Nicola Santoro, December 1982
SCS-TR-13	**On The Message Complexity of Distributed Problems** Nicola Santoro, December 1982
SCS-TR-14	**A Common Basis for Similarity Measures Involving Two Strings** R.L. Kashyap and B.J. Oommen, January 1983

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-15 **Similarity Measures for Sets of String**
R.L. Kashyap and B.J. Oommen, January 1983

SCS-TR-16 **The Noisy Substring Matching Problem**
R.L. Kashyap and B.J. Oommen, January 1983

SCS-TR-17 **Distributed Election in a Circle Without a Global Sense of Orientation**
E. Korach, D. Rotem, and N. Santoro, January 1983

SCS-TR-18 **A Geometrical Approach to Polygonal Dissimilarity and the Classification of Closed Boundaries**
R.L. Kashyap and B.J. Oommen, January 1983

SCS-TR-19 **Scale Preserving Smoothing of Polygons**
R.L. Kashyap and B.J. Oommen, January 1983

SCS-TR-20 **Not-Quite-Linear Random Access Memories**
Jim des Rivieres, Wilf R. LaLonde, and Mike Dixon, March 1983

SCS-TR-21 **Shout Echo Selection in Distributed Files**
D. Rotem, N. Santoro, J.B. Sydney, March 1983.

SCS-TR-22 **Distributed Ranking**
E. Korach, D. Rotem, N. Santoro, March 1983.

SCS-TR-23 **A Reduction Technique for Selection in Distributed Files : I**
N. Santoro, J.B. Sidney, April 1983.
Replaced by SCS-TR-69

SCS-TR-24 **Learning Automata Possessing Ergodicity of the Mean : The Two Action Case**
M.A.L. Thathachar and B.J. Oommen, May 1983.

SCS-TR-25 **Actors - The Stage is Set**
John R. Pugh, June 1983.

SCS-TR-26 **On the Essential Equivalence of Two Families of Learning Automata**
M.A.L. Thathachar and B.J. Oommen, May 1983.

SCS-TR-27 **Generalized Krylov Automata and their Applicability to Learning in Nonstationary Environments**
B.J. Oommen, June 1983.

SCS-TR-28 **Actor Systems: Selected Features**
out-of-print
Wilf R. LaLonde, July 1983.

SCS-TR-29 **Another Addendum to Kronecker's Theory of Pencils**
M.D. Atkinson, August 1983.

SCS-TR-30 **Some Techniques for Group Character Reduction**
M.D. Atkinson and R.A. Hassan, August 1983.

SCS-TR-31 **An Optimal Algorithm for Geometrical Congruence**
M.D. Atkinson, August 1983.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-32 **Multi-Action Learning Automata Possessing Ergodicity of the Mean**
B.J. Oommen and M.A.L. Thathachar, August 1983.

SCS-TR-33 **Fibonacci Graphs, Cyclic Permutations and Extremal Points**
Out-of-print N. Santoro and J. Urrutia, December 1983

SCS-TR-34 **Distributed Sorting**
D. Rotem, N. Santoro, and J.B. Sydney, December 1983.

SCS-TR-35 **A Reduction Technique for Selection in Distributed Files: II**
N. Sanatoro, M. Scheutzow, and J.B. Sidney, December 1983.

SCS-TR-36 **The Asymptotic Optimality of Discretized Linear Reward-Inaction Learning Automata**
B.J. Oommen and Eldon Hansen, January 1984.

SCS-TR-37 **Geometric Containment is Not Reducible to Pareto Dominance**
N. Santoro, J.B. Sidney, S.J. Sidney, and J. Urrutia, January 1984.

SCS-TR-38 **An Improved Algorithm for Boolean Matrix Multiplication**
N. Santoro and J. Urrutia, January 1984.

SCS-TR-39 **Containment of Elementary Geometric Objects**
J. Sack, N. Santoro and J. Urrutia, February 1984

SCS-TR-40 **SADE: A Programming Environment for Designing and Testing Systolic Algorithms**
J.P. Corriveau and N. Santoro, February 1984.

SCS-TR-41 **Intersection Graphs, $\{B_1\}$-Orientable Graphs and Proper Circular Arc Graphs**
Jorge Urrutia, February 1984.

SCS-TR-42 **Minimum Decompositions of Polygonal Objects**
J. Mark Keil and Jorg-R. Sack, March 1984.

SCS-TR-43 **An Algorithm for Merging Heaps**
Jorg-R. Sack and Thomas Strothotte, March 1984.

SCS-TR-44 **A Digital Hashing Scheme for Dynamic Multiattribute Files**
E.J. Otoo, March 1984

SCS-TR-45 **Symmetric Index Maintenance Using Multidimensional Linear Hashing**
E.J. Otoo, March 1984

SCS-TR-46 **A Mapping Function for the Directory of a Multidimensional Extendible Hashing**
E.J. Otoo, March 1984.

SCS-TR-47 **Translating Polygons in the Plane**
Jorg-R. Sack, March 1984.

SCS-TR-48 **Constrained String Editing**
J. Oommen, May 1984.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-49
Out-of-print
O(N) Election Algorithms in Complete Networks with Global Sense of Orientation
Jorg Sack, Nicola Santoro, Jorge Urrutia, May 1984

SCS-TR-50
____
The Design of a Program Editor Based on Constraints
Christopher A. Carter and Wilf R. LaLonde, May 1984.

SCS-TR-51
____
Discretized Linear Inaction-Penalty Learning Automata
B.J. Oommen and Eldon Hansen, May 1984.

SCS-TR-52
____
Sense of Direction, Topological Awareness and Communication Complexity
Nicola Santoro, May 1984.

SCS-TR-53
____
Optimal List Organizing Strategy Which Uses Stochastic Move-to-Front Operations
B.J. Oommen, June 1984.

SCS-TR-54
____
Rectilinear Computational Geometry
J. Sack, June 1984.

SCS-TR-55
____
An Efficient, Implicit Double-Ended Priority Queue
M.D. Atkinson, Jorg Sack, Nicola Santoro, T. Strothotte, July 1984.

SCS-TR-56
____
Dynamic Multipaging: A Multidimensional Structure for Fast Associative Searching
E. Otoo, T.H. Merrett, August 1984.

SCS-TR-57
____
Specialization, Generalization and Inheritance
Wilf R. LaLonde, John R. Pugh, August 1984.

SCS-TR-58
____
Computer Access Methods for Extendible Arrays of Varying Dimensions
E. Otoo, August 1984.

SCS-TR-59
____
Area-Efficient Embeddings of Trees
J.P. Corriveau, Nicola Santoro, August 1984.

SCS-TR-60
____
Uniquely Colourable m-Dichromatic Oriented Graphs
V. Neumann-Lara, N. Santoro, J. Urrutia, August 1984.

SCS-TR-61
____
Analysis of Distributed Algorithms for Extrema Finding in a Ring
D. Rotem, E, Korach and N. Santoro, August 1984.

SCS-TR-62
out-of-print
On Zigzag Permutations and Comparisons of Adjacent Elements
M.D. Atkinson, October 1984. See Information Processing Letters 21 ('85) 187-189

SCS-TR-63
____
Sets of Integers with Distinct Differences
M.D. Atkinson, A. Hassenklover, October 1984.

SCS-TR-64
____
Teaching Fifth Generation Computing: The Importance of Small Talk
Wilf R. LaLonde, Dave A. Thomas, John R. Pugh, October 1984.

SCS-TR-65
____
An Extremely Fast Minimum Spanning Circle Algorithm
B.J. Oommen, October 1984.

SCS-TR-66
____
On the Futility of Arbitrarily Increasing Memory Capabilities of Stochastic Learning Automata
B.J. Oommen, October 1984. Revised May 1985.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-67 ________	**Heaps in Heaps** T. Strothotte, J.-R. Sack, November 1984. Revised April 1985.
SCS-TR-68 out-of-print	**Partial Orders and Comparison Problems** M.D. Atkinson, November 1984. See Congressus Numerantium 47 ('86), 77-88
SCS-TR-69 ________	**On the Expected Communication Complexity of Distributed Selection** N. Santoro, J.B. Sidney, S.J. Sidney, February 1985.
SCS-TR-70 ________	**Features of Fifth Generation Languages: A Panoramic View** Wilf R. LaLonde, John R. Pugh, March 1985.
SCS-TR-71 ________	**Actra: The Design of an Industrial Fifth Generation Smalltalk System** David A. Thomas, Wilf R. LaLonde, April 1985.
SCS-TR-72 ________	**Minmaxheaps, Orderstatisticstrees and their Application to the Coursemarks Problem** M.D. Atkinson, J.-R. Sack, N. Santoro, T. Strothotte, March 1985.
SCS-TR-73 ________	**Designing Communities of Data Types** Wilf R. LaLonde, May 1985.
SCS-TR-74 ________	**Absorbing and Ergodic Discretized Two Action Learning Automata** B. John Oommen, May 1985.
SCS-TR-75 ________	**Optimal Parallel Merging Without Memory Conflicts** Selim Akl and Nicola Santoro, May 1985
SCS-TR-76 ________	**List Organizing Strategies Using Stochastic Move-to-Front and Stochastic Move-to-Rear Operations** B. John Oommen, May 1985.
SCS-TR-77 ________	**Linearizing the Directory Growth in Order Preserving Extendible Hashing** E.J. Otoo, July 1985.
SCS-TR-78 ________	**Improving Semijoin Evaluation in Distributed Query Processing** E.J. Otoo, N. Santoro, D. Rotem, July 1985.
SCS-TR-79 ________	**On the Problem of Translating an Elliptic Object Through a Workspace of Elliptic Obstacles** B.J. Oommen, I. Reichstein, July 1985.
SCS-TR-80 ________	**Smalltalk - Discovering the System** W. LaLonde, J. Pugh, D. Thomas, October 1985.
SCS-TR-81 ________	**A Learning Automation Solution to the Stochastic Minimum Spanning Circle Problem** B.J. Oommen, October 1985.
SCS-TR-82 ________	**Separability of Sets of Polygons** Frank Dehne, Jörg-R. Sack, October 1985.
SCS-TR-83 ________	**Extensions of Partial Orders of Bounded Width** M.D. Atkinson and H.W. Chang, November 1985.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-84 **Deterministic Learning Automata Solutions to the Object Partitioning Problem**
B. John Oommen, D.C.Y. Ma, November 1985

SCS-TR-85 **Selecting Subsets of the Correct Density**
M.D. Atkinson, December 1985

SCS-TR-86 **Robot Navigation in Unknown Terrains Using Learned Visiblity Graphs. Part I: The Disjoint Convex Obstacles Case**
B. J. Oommen, S.S. Iyengar, S.V.N. Rao, R.L. Kashyap, February 1986

SCS-TR-87 **Breaking Symmetry in Synchronous Networks**
Greg N. Frederickson, Nicola Santoro, April 1986

SCS-TR-88 **Data Structures and Data Types: An Object-Oriented Approach**
John R. Pugh, Wilf R. LaLonde and David A. Thomas, April 1986

SCS-TR-89 **Ergodic Learning Automata Capable of Incorporating Apriori Information**
B. J. Oommen, May 1986

SCS-TR-90 **Iterative Decomposition of Digital Systems and Its Applications**
Vaclav Dvorak, May 1986.

SCS-TR-91 **Actors in a Smalltalk Multiprocessor: A Case for Limited Parallelism**
Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986

SCS-TR-92 **ACTRA - A Multitasking/Multiprocessing Smalltalk**
David A. Thomas, Wilf R. LaLonde, and John R. Pugh, May 1986

SCS-TR-93 **Why Exemplars are Better Than Classes**
Wilf R. LaLonde, May 1986

SCS-TR-94 **An Exemplar Based Smalltalk**
Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986

SCS-TR-95 **Recognition of Noisy Subsequences Using Constrained Edit Distances**
B. John Oommen, June 1986