INTERSECTION GRAPHS, $\{B_1\}$-ORIENTABLE GRAPHS AND PROPER CIRCULAR ARC GRAPHS

Jorge Urrutia

SCS-TR-41
January 1984

Centro de Investigacion en Matematicas, Guanajuato, Mexico and School of Computer Science, Carleton University, Canada.

INTERSECTION GRAPHS, $\{B_1\}$-ORIENTABLE GRAPHS AND PROPER CIRCULAR ARC GRAPHS

BY

JORGE URRUTIA

ABSTRACT

A graph G is called a $\{B_1\}$-orientable graph if there exists an orientation $\vec{G}$ of G such that if $x \to y$ and $z \to y$ in $\vec{G}$ then $\{x,z\}$ is an edge of G. In this paper we obtain a characterization of $\{B_1\}$-orientable graphs. A recognition algorithm for $\{B_1\}$-orientable graphs is also presented. A simple recognition algorithm for proper circular arc graphs is then obtained. Finally a new family of $\{B_1'\}$-orientable graphs is constructed.

Centro de Investigación en Matemáticas, Guanajuato, México
and School of Computer Science, Carleton University, Canada.

1: INTRODUCTION

Some families of intersection graphs such as interval graphs, circular arc graphs and triangulated graphs can be characterized by using certain special orientations on them. Some of these orientations have been studied in various papers (see [3], [4], [5], [6], [7], [9], [11]).

In particular, let B_1 and B_2 be the directed graphs shown in Figure 1. An orientation $\vec{G}$ of a graph G is a $\{B_1\}$-orientation (or $\{B_2\}$, $\{B_1,B_2\}$) if $\vec{G}$ does not contain B_1 as an induced subgraph (respectively B_2, B_1 and B_2 simultaneously).

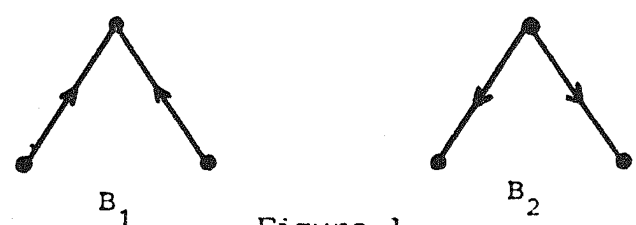

Figure 1

Characterizations of interval graphs, proper interval graphs, triangulated graphs and proper circular arc graphs can be obtained in terms of $\{B_1\}$, $\{B_1,B_2\}$, $\{B_1\}^*$ and $\{B_1,B_2\}^*$ orientations (where $\{B_1\}^*$ and $\{B_1,B_2\}^*$ are acyclic $\{B_1\}$ or $\{B_1,B_2\}$ orientations respectively).

The objective of this paper is to give a characterization of $\{B_1\}$-orientable graphs. An algorithm to find $\{B_1,B_2\}$-orientations (when they exist) is also obtained. This characterization of $\{B_1\}$-orientable graphs completes Table II of [9].

2: CHARACTERIZATION OF $\{B_1\}$-ORIENTABLE GRAPHS.

Let {a,b} be an edge of a graph G . An edge {u,v} is (b,u)-strongly connected to {a,b} if there exists a path P={a,b}, $\{b,v_1\}$, ... , $\{v_m,u\}$,{u,v} such that P has no triangular chords. Clearly if {a,b} and {u,v} are (b,u)-strongly connected they are (u,v)-strongly connected as well.

Notice that in any $\{B_1\}$-orientation of G in which a->b, the edge {u,v} must be oriented u->v, i.e. a->b forces u->v.

Furthermore if {a,b} is such that it is (b,u) and (b,v) strongly connected to {u,v}, then there exists no {B1}-orientation of G in which a->b. For if a->b in a {B1}-orientation of G, then u->v since {u,v} is (b,u)-strongly connected to {a,b}. Similarly v->u; since this is not possible, then in any {B1}-orientation of G, b->a.

We say that an edge {a,b} is (a->b){B1}-admissible if there exists no edge {u,v} of G that is (b,u) and (b,v)-strongly connected to {a,b}. Finally we call an edge {a,b} a {B_1}-admissible edge if it is (a->b) or (b->a){B1}-admissible.

Notice that there are edges of G that are (a->b) but not (b->a){B_1}-admissible; while others are (a->b) and (b->a){B_1}-admissible (see Figure 2).

Figure 2

We shall prove the following theorem which characterizes {B_1}-orientable graphs.

THEOREM 1: A graph G is {B_1}-orientable if and only if all edges of G are {B_1} admissible.

Proof: Clearly if G is {B_1}-orientable then every edge of G is {B_1}-admissible, for if a->b in a {B1}-orientation of G, then {a,b} is (a->b){B_1} admissible.

Let us choose a minimal set S consisting of m edges e ,. ...,e of G satisfying the following conditions:

a) S = { e_i={a_i,b_i} | (a_i->b_i) are {B_1} admissible; $e_i \neq e_j$ ($i \neq j$) i=1,2,....,m } (Notice that a_i could be the same as a_j or b_j for some $j \neq i$).

b) Every edge {u,v} of G is either (b_i,u) or (b_i,v)-strongly connected to e_i={a_i,b_i} for at least one element of S.

c) {a_i,b_i} is not (b_j,b_i) or (b_j,a_i)-strongly connected to {a_j,b_j} ($j<i$).

Such a set S can be constructed inductively as follows:

i) Choose any edge $e_1=\{a_1,b_1\}$ of G which is $(a_1\to b_1)\{B_1\}$ admissible. Set $S_1=\{\{a_1,b_1\}\}$.

ii) Suppose we already have S_i for some $i>1$. Two cases arise:

(a') Every edge $\{u,v\}$ of G is (b_j,u) or (b_j,v)-strongly connected to some e_j element of S_i. In this case S_i satisfies a), b) and c) as above. Set $S=S_i$.

(b') There exists a $(a_{i+1}\to b_{i+1})\{B_1\}$-admissible edge e_{i+1} of G not (b_j,a_{i+1}) or (b_j,b_{i+1})-strongly connected to any $e_j\in S_i$. In this case we define $S_{i+1}=S\cup\{e_{i+1}\}$.

By using S we can define an orientation $\vec{G}$ of G as follows:

i') Orient $a_i\to b_i$ $1\leq i\leq m$.

ii') If $\{u,v\}$ is (b_i,u)-strongly connected to $\{a_i,b_i\}$ for some i, orient $u\to v$; otherwise orient $v\to u$.

We claim that $\vec{G}$ is well defined and also a $\{B_1\}$-orientation of G. For if there is an edge $\{u,v\}$ such that for $i\neq j$ $(i<j)$ $\{u,v\}$ is (b_i,u) and (b_j,v)-strongly connected to $\{a_i,b_i\}$ and $\{a_j,b_j\}$ respectively, then $\{a_j,b_j\}$ is (b_j,b_i)-strongly connected to $\{a_i,b_i\}$; which is a contradiction.

Suppose now that $\vec{G}$ is not a $\{B_1\}$-orientation of G. Then there exist x, y, z, vertices of G such that $\{x,y\}$ is not an edge of G but $x\to z$ and $y\to z$ in $\vec{G}$.

However this is not possible for if $x\to z$ there is an edge $\{a_i,b_i\}$ of S such that $\{x,z\}$ is (b_i,x)-strongly connected to $\{a_i,b_i\}$. Therefore $z\to y$ in G, a contradiction. Therefore $\vec{G}$ is a $\{B_1\}$-orientation of G.

2:2 A RECOGNITION ALGORITHM FOR $\{B_1\}$-ORIENTABLE GRAPHS.

The proof of Theorem 1 suggests the following algorithm for recognizing $\{B_1\}$-orientable graphs.

ALGORITHM 1

Given an unoriented graph proceed as follows:

STEP 1: Choose an unoriented edge $\{x,y\}$. Label it $\{a,b\}$.

STEP 2: Orient a->b and colour this edge.

STEP 3: Find all edges {b,v} such that {a,v} is not an edge of G. If at least one of them is already oriented v->b, go to STEP 5. Else colour and orient all UNORIENTED edges {b,v} from b->v. (At this stage some edges {b,v} may be already oriented and possibly are uncoloured; DO NOT COLOUR THEM.) Mark a->b.

STEP 4: Find an unmarked oriented and coloured edge u->v; label it a->b and go to STEP 3. If all coloured oriented edges are already marked, then uncolour and unmark all coloured and marked edges. Set BOOL=TRUE and go to STEP 6.

STEP 5 : If BOOL=FALSE go to STEP 7.
If BOOL=TRUE, then BOOL=FALSE.
Unorient and unmark all coloured oriented edges; relabel {x,y} as {b,a}, and go to STEP 2.

STEP 6: If all edges of G are already oriented then G is $\{B_1\}$-oriented; go to STEP 8. Else go to STEP 2.

STEP 7: G is not $\{B_1\}$-orientable.

STEP 8: STOP.

It can be easily proved that Algorithm 1 has complexity $O(\Delta \cdot |E(G)|^2)$. The correctness of Algorithm 1 follows immediately from the "only if" part of the proof of Theorem 1.

We shall finish this section by giving an interesting property of $\{B_1\}$-orientable graphs. By using this property, we will be able to systematically modify any $\{B_1\}$-orientation $\vec{G}$ of G to obtain any other $\{B_1\}$-orientation $\vec{G}'$ of G by means of some very simple changes. Let $\vec{G}$ be a $\{B_1\}$-orientation of G and {a,b} an edge of G, oriented b->a in G. If {a,b} is (a->b)$\{B_1\}$-admissible, let $\vec{G}$(a->b) be the oriented graph obtained from $\vec{G}$ by reorienting u->v all edges {u,v} of G which are (b,u)-strongly connected to {a,b}. (See Figure 3.)

LEMMA 1: Let $\vec{G}$ be a $\{B_1\}$-orientation of a graph G and {a,b} a (a->b)-admissible edge of G. Then $\vec{G}$(a->b) is also a $\{B_1\}$-orientation of G.

Proof: Suppose that $\vec{G}(a\to b)$ is not a $\{B_1\}$-orientation of G. Then there exist x, y, z, vertices of G, such that $x\to z$, $y\to z$ in $\vec{G}(a\to b)$ but $\{x,y\}$ is not an edge of G. Clearly $\{x,z\}$ is (b,x)-strongly connected to $\{a,b\}$ or $\{y,z\}$ is (b,y)-strongly connected to $\{a,b\}$. However if $\{x,z\}$ is (b,x) strongly connected to $\{a,b\}$ then $\{z,x\}$ is (b,z) strongly connected to $\{a,b\}$. Hence $\{z,y\}$ must be oriented $z\to y$ in $G(a\to b)$ which contradicts $y\to z$. Similarly if $\{y,z\}$ is (b,y) strongly connected to $\{a,b\}$ then $z\to x$ in $\vec{G}(a\to b)$; therefore $\vec{G}(a\to b)$ is a $\{B_1\}$-orientation of G.

A generalization of Lemma 1 will be given in the next section. This generalization will allow us to obtain a recognition algorithm for $\{B_1,B_2\}$-orientable graphs.

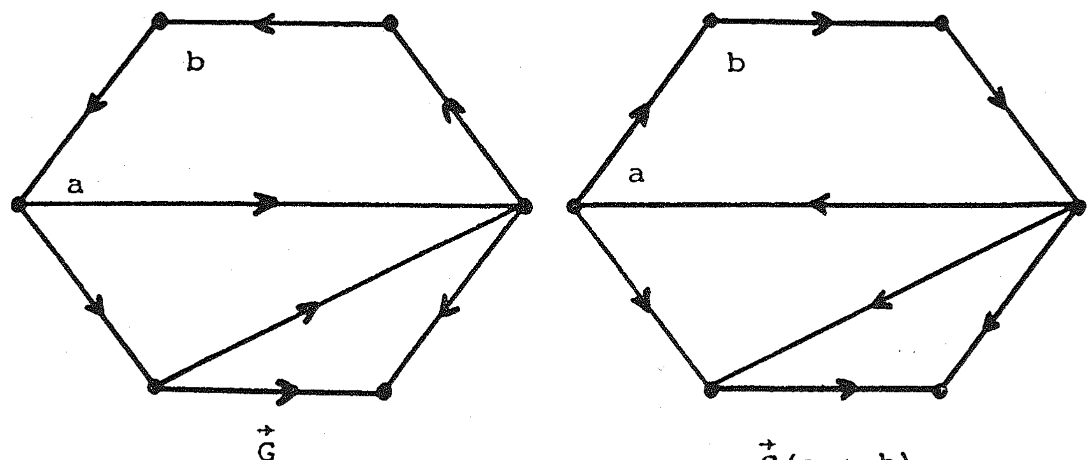

Figure 3.

3: RECOGNIZING $\{B_1,B_2\}$-ORIENTABLE GRAPHS.

In this section we present a $O(\Delta\cdot|E(G)|)$ algorithm to recognize $\{B_1,B_2\}$-orientable graphs. This produces a $O(\Delta\cdot|E(G)|)$ recognition of proper circular arc graphs. Unlike many recognition algorithms of circular arc graphs, this algorithm is very simple and easy to understand.

Let $\vec{G}$ be a $\{B_1,B_2\}$-orientation of a graph G and $a\to b$ an edge of $\vec{G}$. Let $\vec{G}(b\to a)$ be the oriented graph obtained from G by reorienting $b\to a$ and all edges $\{u,b\}$ of G which are strongly connected to $\{a,b\}$, (i.e. they are (x,y)-strongly connected, where x is either a or b and y is any of u or v). Notice that these edges are the edges we are forced to reorient in $\vec{G}$ if we want to maintain a $\{B_1,B_2\}$-orientation of G after reorienting $b\to a$.

The following result is a generalization of Lemma 1.

LEMMA 2: If $\vec{G}$ is a $\{B_1,B_2\}$-orientation of G, then $\vec{G}$ (b->a) is also a $\{B_1,B_2\}$-orientation of G.

Proof: The proof is similar to that of Lemma 1.

Remark 1: An immediate consequence of Lemma 1 is that if a graph G is a $\{B_1,B_2\}$-orientable graph and {u,v} is an edge of G, then there always exists a $\{B_1,B_2\}$-orientation of G in which u->v.

An immediate consequence of Lemma 2 is the following recognition algorithm of $\{B_1,B_2\}$-orientable graphs.

ALGORITHM 2

Given a graph G, proceed as follows:

STEP 1: Choose an unoriented edge {x,y}. Label it {a,b}. If no such edge exists, go to STEP 6.

STEP 2: Orient a->b.

STEP 3: Find all edges {u,a} such that {a,b} is not an edge of G. If at least one of these edges is oriented a->u, go to STEP 7. Else orient them u->a.

STEP 4: Find all edges {b,u} such that {a,u} is not an edge of G. If at least one of them is oriented u->b, go to STEP 7. Else orient them b->u.

Mark the edge a->b.

STEP 5: Find an unmarked oriented edge x->y and relabel it a->b. If no such edge exists, go to STEP 1. Else go to STEP 3.

STEP 6: The graph G has been successfully $\{B_1,B_2\}$-oriented. STOP.

STEP 7: The graph G is not $\{B_1,B_2\}$-orientable. STOP.

Clearly the running time of ALGORITHM 2 is dominated by steps three and four, each of which takes at most Δ operations. Since these steps are repeated once for each edge of G, the complexity of ALGORITHM 2 is $O(\Delta \cdot |E(G)|)$.

1: CONSTRUCTING $\{B_1\}$-ORIENTABLE GRAPHS.

It can be easily shown that interval graphs, triangulated graphs and circular arc graphs are all $\{B_1\}$-orientable graphs. The converse is not true, there are graphs which are $\{B_1\}$-orientable but do not belong to any of these families. The graph in Figure 4 is one of such graphs. However it can be represented as the intersection graph of a set of curves on the plane.

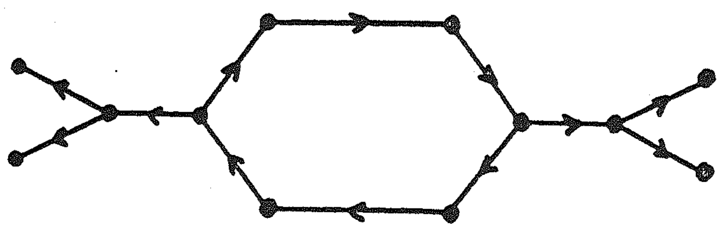

Figure 4.

A wider family of $\{B_1\}$-orientable graphs which contains all the families previously mentioned can be obtained as follows:

Let $\vec{G}$ be an orientation of a connected graph G such that $d^-(v)\leq 1$ for every vertex v of G. Let $F=\{T_1, T_2, \ldots, T_k\}$ be a family of subtrees of G such that no point of G is an end point of more than one subtree of F. Each T_i $(1\leq i\leq k)$ inherits an orientation $\vec{T}_i$ of G such that T_i has exacty one source S_i . (This is a consequence of the restriction $d^-(v)\leq 1$ of the orientation $\vec{G}$).

If in addition we require that $T_i \cap T_j$ always be a subtree of G, it can be easily shown that if $T_i \cap T_j \neq \emptyset$ then $S_i \in T_j$ or $S_j \in T_i$. Let H be the intersection graph of F, i.e. $V(H)=F$ and $E(H)=\{ \{T_i, T_j\} \mid T_i \cap T_j \neq \emptyset \}$.

A $\{B_1\}$-orientation of H can now be obtained as follows: an edge $\{T_i, T_j\}$ of H is oriented $T_i \rightarrow T_j$ if $S_j \in T_i$; otherwise $T_j \rightarrow T_i$.

Clearly this is a $\{B_1\}$-orientation of H, since if $T_i \rightarrow T_j$ and $T_k \rightarrow T_j$ then S_j belongs to T_i and T_k. Therefore $\{T_i, T_j\}$ is an edge of H.

Notice that if G has n vertices, then G has at most n edges. Then G contains at most one cycle. When G has exactly n edges, G is a path or a tree. In the first case H will be an interval graph and in the second H is a triangulated graph. When G has n edges, G is a cycle or a tree plus an extra edge In the first case H is a circular arc graph. For the second case we obtain a new class of graphs that are $\{B_1\}$-orientable.

There are, however, many $\{B_1\}$-orientable graphs which can not be obtained by using the previous method (see Figure 5). However all $\{B_1\}$-orientable graphs known to us are intersection graphs of sets of curves on the plane. We venture the following conjecture.

CONJECTURE 1: If a graph G is $\{B_1\}$-orientable then G is the intersection graph of some family of curves on the plane.

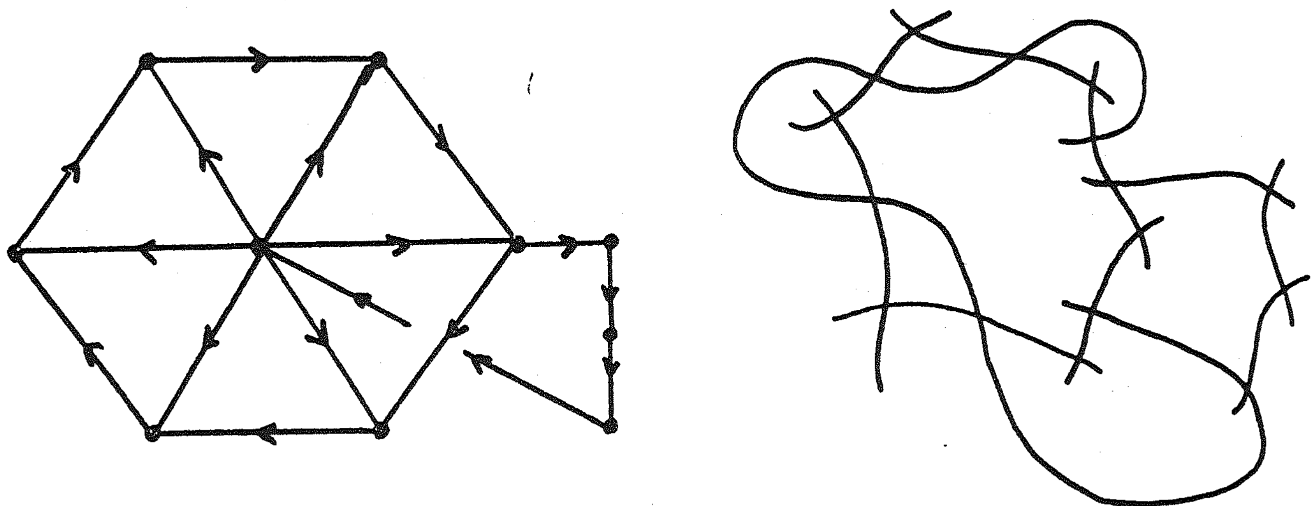

Figure 5.

Acknowledgements: We would like to thank Dr. Martin C. Golumbic for calling our attention to the characterization problem of $\{B_1\}$-orientable graphs.

R E F E R E N C E S

[1] G. Ehrlich, S. Even and R. E. Tarjan, Intersection Graphs of Curves in the Plane. J. Combinatorial Theory (B) 21 (1976) 8-20.

[2] F. Gavril, Algorithms For a Maximum Clique and a Maximum Independent Set of a Circle Graph. Networks 3 (1973) 261-273

[3] P. C. Gilmore and A. J. Hoffman. A Characterization of Comparability Graphs and of Interval Graphs. Canad. J. Math. 16 (1962) 1370-1381

[4] M. C. Golumbic. Algorithmic Graph Theory and Perfect Graphs Academic Press, New York, 1980, MR 81e:68081.

[5] A. Pnueli, A. Lempel and S. Even. Transitive Orientations of Graphs and Identification of Permutation Graphs. Canad. J. Math. 23 (1971) 160-175.

[6] R. C. Read, D. Rotem and J. Urrutia. Orientations of Circle Graphs. J. of Graph Theory 6 (1982) 325-341.

[7] D. J. Rose. Triangulated Graphs and the Elimination Process. J. Math. Anal. Appl. 32 (1970) 597-602.

[8] D. Rotem and J. Urrutia. Finding Maximum Cliques in Circle Graphs. Networks 11 (1981) 269-278.

[9] D. J. Skrien. a Relationship Between Triangulated Graphs, Comparability Graphs, Proper Interval Graphs, Proper Circular Arc Graphs and Nested Interval Graphs. J. of Grapn Theory 6 (1982) 309-316.

[10] A. C. Tucker. Structure Theorems for Some Circular Arc Graphs. Discrete Math. 7 (1974) 167-195.

[11] J. Urrutia. Intersection Graphs of Some Families of Plane Curves. Ph. D. Thesis. Univ. of Waterloo (1980).

CARLETON UNIVERSITY

School of Computer Science

BIBLIOGRAPHY OF SCS TECHNICAL REPORTS

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**
Nicola Santoro, Ramez Khatib, July 1982.

SCS-TR-7 — **CONCURRENCY CONTROL IN LARGE COMPUTER NETWORKS**
Nicola Santoro, Hasan Ural, July 1982.

SCS-TR-8 Out of Print — **ORDER STATISTICS ON DISTRIBUTED SETS**
Nicola Santoro, Jeffrey B. Sidney, July 1982.

SCS-TR-9 — **OLIGARCHICAL CONTROL OF DISTRIBUTED PROCESSING SYSTEMS**
Moshe Krieger, Nicola Santoro, August 1982.

SCS-TR-10 — **COMMUNICATION BOUNDS FOR SELECTION IN DISTRIBUTED SETS**
Nicola Santoro, Jeffrey B. Sidney, September 1982.

SCS-TR-11 — **SIMPLE TECHNIQUE FOR CONVERTING FROM A PASCAL SHOP TO A C SHOP**
Wilf R. LaLonde, 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.

SCS-TR-15 **SIMILARITY MEASURES FOR SETS OF STRINGS**
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, 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 LaLonde and Mike Dixon, August 1982, Revised March 1, 1983.

SCS-TR-21 **SHOUT ECHO SELECTION IN DISTRIBUTED FILES**
D. Rotem, N. Santoro, J. B. Sidney, 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 ENVIROMENTS**
B. J. Oommen, June 1983.

SCS-TR-28 **ACTOR SYSTEMS: SELECTED FEATURES**
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.

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**
N. Santoro and J. Urrutia, December 1983

SCS-TR-34 **DISTRIBUTED SORTING**
D. Rotem, N. Santoro, and J. B. Sidney, December 1983

SCS-TR-35 **A REDUCTION TECHNIQUE FOR SELECTION IN DISTRIBUTED FILES: II**
N. Santoro, 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 AGLORITHM 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 }-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

SCS-TR-49 **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 **RECTINLINEAR 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

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 A DISTRIBUTED ALGORITHMS FOR EXTREMA FINDING IN A RING**
D. Rotem, E. Korach and N. Santoro, August 1984.

SCS-TR-62 **ON ZIGZAG PERMUTATIONS AND COMPARISONS OF ADJACENT ELEMENTS**
M. D. Atkinson, October 1984

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.

SCS-TR-67 **HEAPS IN HEAPS**
T. Strothotte, J.-R. Sack, November 1984. Revised April 1985.

SCS-TR-68 **PARTIAL ORDERS AND COMPARISON PROBLEMS**
M. D. Atkinson, November 1984.

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, Th. 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.