TOURNAMENT-LIKE ORIENTED GRAPHS

by

Jing Huang

M.Sc., Institute of System Sciences, Academia Sinica, China, 1986

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the Department
of
Mathematics and Statistics

SIMON FRASER UNIVERSITY
August, 1992

ISBN 0-315-83643-1

APPROVAL

Name: Jing Huang

Degree: Doctor of Philosophy

Title of thesis: **Tournament-like Oriented Graphs**

Examining Committee: Dr. Alistair H. Lachlan, Professor
Chair

Dr. Pavol Hell, Professor
Senior Supervisor

Dr. Brian Alspach, Professor

Dr. Katherine Heinrich, Professor

Dr. Arvind Gupta, Assistant Professor

Dr. Derek G. Corneil, Professor
External Examiner
Department of Computer Science
University of Toronto

Date Approved: August 31, 1992

PARTIAL COPYRIGHT LICENSE

Title of Thesis/Project/Extended Essay

Tournament-like Oriented Graphs

Author:

(signature)

Jing Huang

(name)

Aug. 12 1992

(date)

Abstract

A local tournament is an oriented graph in which the inset as well as the outset of each vertex induces a tournament. Local tournaments possess many properties of tournaments and have interesting structure. In 1982, Skrien proved (in different terminology), using a deep structural characterization of proper circular arc graphs by Tucker, that a connected graph is local-tournament-orientable if and only if it is a proper circular arc graph.

In Chapter 2, we shall give a simple $O(m\Delta)$ algorithm to decide if a graph can be oriented as a local tournament, and hence whether or not it is a proper circular arc graph. We analyze relationships among local tournaments, local transitive tournaments, and proper circular arc graphs. We obtain theorems to describe all possible local-tournament orientations of a proper circular arc graph.

In Chapter 3, we shall present an $O(m\Delta)$ algorithm to recognize comparability graphs and to calculate transitive orientations. Our method can be applied to recognize proper circular arc graphs and to find local-transitive-tournament orientations, and can also be applied to recognize proper interval graphs and to find acyclic local-tournament orientations. We shall give a simple proof of Skrien's theorem, which does not depend on Tucker's result.

In Chapter 4, we shall present two $O(m+n)$ time algorithms. One is for recognizing proper interval graphs and for finding an associated interval family. The other is for recognizing proper circular arc graphs and for finding an associated circular arc family.

In Chapter 5, we shall obtain two additional $O(m+n)$ time algorithms for proper circular arc graphs by using the auxiliary local-tournament orientations. One is for finding maximum cliques, and the other is for determining c-colourablity.

In Chapter 6, we shall introduce a new class of oriented graphs namely, in-tournaments, which contains the class of local tournaments. We shall show that some of the basic and very nice properties of tournaments extend not only to local tournaments, but also to this more general class of digraphs. Our results imply a polynomial time algorithm for finding hamiltonian paths and cycles in the class of in-tournaments. We shall also investigate the the class of graphs which are orientable as in-tournaments.

Finally, in Chapter 7, we shall introduce another class of oriented graphs, i.e., those of Moon type. We shall find a close relationship between the class of oriented graphs of Moon type and the class of local tournaments. In fact, oriented graphs of Moon type can be characterized in terms of local transitive tournaments.

Acknowledgements

I would like to express my sincere gratitude to my Senior Supervisor, Dr. Pavol Hell, for his encouragement, guidance, and patience during the preparation of this thesis. I consider it an honor and priviledge to have had the opportunity to work with him these past few years.

I must thank my wife for her unflagging support over the years.

Dedication

To my mother .....

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

A tournament is a complete oriented graph. Tournaments have been a popular object of study since the early days of graph theory. There is now an extensive theory associated with tournaments, [13, 57]. A semicomplete digraph is obtained from a tournament by adding additional arcs, i.e., it is a digraph in which any two distinct vertices are joined by at least one arc. Clearly, semicomplete digraphs generalize tournaments. Many difficult problems for general digraphs can be easily solved for tournaments and semicomplete digraphs. For instance, the problems of finding a hamiltonian path and finding a hamiltonian cycle are NP-complete for general digraphs, cf. [50], and polynomial for both tournaments and semicomplete digraphs, cf. [59].

It is natural to look for a larger class of digraphs, which still allows as many problems to remain tractable as possible. Recently in [4], Bang-Jensen introduced one such interesting class of digraphs. He called them locally semicomplete digraphs. A locally semicomplete digraph is a digraph in which the outset as well as the inset of each vertex is semicomplete. In [4], Bang-Jensen proved that most properties that hold for semicomplete digraphs also hold for locally semicomplete digraphs. For instance, a connected locally semicomplete digraph has a directed hamiltonian path.

A strong locally semicomplete digraph has a directed hamiltonian cycle. Moreover there are polynomial algorithms to find such a path or a cycle.

An oriented graph is a local tournament if the outset as well as the inset of each vertex is a tournament, [20, 39]. So a local tournament is an oriented graph which is locally semicomplete. A local transitive tournament is an oriented graph in which the outset as well as the inset of each vertex is a transitive tournament, [20, 47]. Local tournaments are natural generalizations of tournaments, and local transitive tournaments are natural generalizations of transitive tournaments.

A graph G is a circular arc graph if there is a one-to-one correspondence between the vertex set of G and a family $\mathcal{F}$ of circular arcs on a circle such that two vertices are adjacent if and only if the corresponding two circular arcs intersect. The family $\mathcal{F}$ is called a circular arc representation of G. If the circular arcs can be chosen so that no one is completely contained in another, then the corresponding graph is a proper circular arc graph. Similarly a graph is an interval graph if there is a one-to-one correspondence between the vertex set and a family $\mathcal{I}$ of intervals on the real line such that two vertices are adjacent if and only if the corresponding two intervals intersect. The family $\mathcal{I}$ is called an interval representation of the graph. Again if the intervals can be chosen so that no one is completely contained in another, then the graph is a proper interval graph. Interval graphs, proper interval graphs, circular arc graphs, and proper circular arc graphs have practical importance in many different sciences (e.g., genetics, archeology, ecology, computer science, electronics), cf. [15, 27, 33, 36, 51, 67, 73].

Local tournaments not only possess many properties of tournaments but also have their own additional structure. In 1982, Skrien obtained a result which implies a relationship between local tournaments and proper circular arc graphs, [71]. Specifically, a connected graph is a proper circular arc graph if and only if it can be oriented as a local tournament. This view leads to a new way to investigate proper circular arc graphs, namely, by studying local tournaments. In fact, as we shall show, many typical problems can be attacked in this way, and solved efficiently.

According to Skrien's result, the problem of testing if a connected graph is a proper circular arc graph is the same as the problem of testing if the graph is orientable as a local tournament. The former problem was solved by Tucker with an $O(n^2)$ time algorithm (all complexities discussed here are worst-case), which makes use of a matrix characterization of proper circular arc graphs (cf. [81]). We shall transform the latter problem to one of testing 2-colourability of an associate graph, which we can solve in time $O(m\Delta)$.

Circular arc graphs and proper circular arc graphs have been extensively studied for over twenty years and many nice results have been obtained for these graphs (cf. [17, 33, 34, 36, 51, 56, 80]). According to the relationship established by Skrien, those results for proper circular arc graphs can be simply transferred to graphs which are orientable as local tournaments.

We observe the following additional relationships: A connected graph is a proper circular arc graph if and only if it is orientable as a local transitive tournament. A graph is a proper interval graph if and only if it is orientable as a non-strong local tournament. Moreover, for a proper circular arc graph, obtaining a local-transitive-tournament orientation is equivalent to finding a corresponding proper circular arc family. Armed with this knowledge, we are able to analyze the structure of local tournaments, and to obtain theorems which describe all possible local-tournament orientations of a proper circular arc graph, and all possible non-strong local-tournament orientations of a proper interval graph. From our theorems, the problem of generating all local tournaments is completely solved.

An oriented graph is locally bicomplete if there is a complete adjacency between the outset and the inset of each vertex, [40]. An oriented graph is transitive if the inset of each vertex dominates the outset of that vertex. A transitively orientable graph is of course local-bicomplete-orientable. A result due to Ghouilà-Houri, reformulated in our terminology, assures that a local-bicomplete-orientable graph is also transitively orientable.

Transitively orientable graphs are also called comparability graphs, [25, 32, 72]. Comparability graphs are an important class of perfect graphs (cf. [33]). The problem of recognizing comparability graphs was first studied by Pnueli, Lempel, and Even, resulting in an algorithm with a time bound of $O(m\Delta)$ (cf. [63]). However this algorithm involves a complicated structural analysis of comparability graphs. We shall provide a simple algorithm by transforming the problem to the problem of testing the local-bicomplete orientability. Our algorithm also finds a local-bicomplete orientation of a comparability graph, in time $O(m\Delta)$.

We shall then introduce a new method which allows us to find a transitive orientation of a comparability graph also in time $O(m\Delta)$. This problem was also solved by Spinrad with an algorithm having a time bound of $O(n^2)$ (cf. [72]). Our method can also be applied to find, in time $O(m\Delta)$, a local-transitive-tournament orientation of a proper circular arc graph, and an acyclic local-tournament orientation of a proper interval graph. As we mentioned above, these orientations are equivalent to proper circular arc representations or proper interval representations. So our method is also useful for these purposes.

There are efficient algorithms to solve various problems for proper circular arc graphs and for general circular arc graphs, provided a circular arc family is given. For instance, the maximum independent set problem, the minimum clique cover problem, and the minimum dominating set problem can all be solved in time $O(n)$ (cf. [46]).

The recognition and the representation problems for circular arc graphs have been solved by Tucker with an $O(n^3)$ time algorithm (cf. [80]). The same problems for proper circular arc graphs were also solved by Tucker, as we mentioned earlier, with an $O(n^2)$ time algorithm, [81]. We shall present an optimal, i.e., $O(m+n)$ time, algorithm to solve the problems for proper circular arc graphs.

For interval graphs, the recognition and the representation problems were first studied by Booth and Lueker (cf. [17]). Their approach led to an $O(m+n)$ time algorithm. However the algorithm obtained by Booth and Lueker involves a complicated data structure called the PQ-tree. For proper interval graphs, we shall give an algorithm of complexity $O(m+n)$ to solve the recognition and the representation problems. Our algorithm makes use of our structure theorems instead of PQ-trees. Recently, Hsu announced an $O(m+n)$ time algorithm for the recognition of interval graphs without using PQ-trees, [44].

Gavril was first to consider the maximum clique problem for circular arc graphs. He solved this problem with an algorithm of complexity $O(n^{\frac{7}{2}})$ which was later improved by Hsu to $O(mn)$ (cf. [43]). For the special case of proper circular arc graphs, we shall give an $O(m+n)$ algorithm to solve the maximum clique problem. If the circular arc representation is given, our algorithm runs in time $O(n \log n)$. The best previous algorithm, due to Apostolico and Hambrusch, which assumes that a circular arc representation is given, has a time bound of $O(n^2 \log \log n)$ (cf. [3]).

The c-colouring problem, NP-complete for circular arc graphs [27], was first shown to be polynomial for proper circular arc graphs by Orlin, Bonuccelli, and Bovet. Their approach consisted of reducing the problem to a shortest path calculation, and resulted in an algorithm with a time bound of $O(n^2)$ (cf. [61]). This algorithm requires also that a circular arc representation be given. Subsequently, other authors improved the algorithm by improving on the shortest path method, culminating in the algorithm of Shih and Hsu, which has a time bound of $O(n^{\frac{3}{2}})$, [70]. Applying our maximum clique algorithm, we are able to solve this problem with a general $O(m+n)$ algorithm, and an $O(n \log n)$ algorithm when a circular arc representation is given.

Note that all of our algorithms may meaningfully be restricted to connected graphs. Then we can replace the complexities $O(m+n)$ by $O(m)$ throughout.

The class of local tournaments can be generalized to the class of in-tournaments, i.e., those oriented graphs in which the inset of each vertex is a tournament, [8, 10, 11]. We shall show that many properties of local tournaments extend to this larger class of oriented graphs. Our results imply a polynomial algorithm for finding hamiltonian paths and cycles. We shall also study those graphs which are orientable as in-tournaments.

A tournament of Moon type is a tournament in which every subtournament is either acyclic or strong. The tournaments of Moon type have been studied by Moon and Guido, [35, 58]. We shall generalize Moon type tournaments to the class of oriented graphs of Moon type, i.e., those oriented graphs in which every connected subgraph is either acyclic or strong. It turns out that there is a close relationship between oriented graphs of Moon type and local tournaments. We shall prove that all oriented graphs of Moon type can be generated from local transitive tournaments by substitution operations.

1.2 Terminology and Notation

A *graph* G is an ordered pair (V, E), where V is a finite set and E is a set of unordered pairs xy with $x, y \in V$ and $x \neq y$. The elements of V are *vertices*, and the elements of E are *edges*. Note that in our definition, we do not allow any loops, i.e., edges joining a vertex to itself. A graph is *simple* if it contains no multiple edges, i.e., edges joining the same two vertices. All graphs are assumed to be simple, unless stated otherwise (cf. Chapter 6). We will use $G = (V, E)$ or simply G to denote a graph.

If xy is an edge, then the vertex x is *adjacent to* the vertex y and is *incident with* the edge xy. We use $x \sim y$ to denote that x is adjacent to y. If x is not adjacent to y, then the vertex x is *non-adjacent* to the vertex y. If $x \sim y$, then y is a *neighbour* of x. The *neighbourhood* of x, denoted by $N(x)$, is the set of all neighbours of x. The number of vertices in $N(x)$ is the *degree* of x, denoted by $deg(x)$. The *maximum degree* of a graph G is the maximum value among the degrees of all vertices of G. We shall use $\Delta(G)$ or simply Δ to denote the maximum degree of G. The *closed neighbourhood* of x, denoted by $N[x]$, is defined to be $N(x) \cup \{x\}$. Note that if two vertices x and y have the same closed neighbourhood, namely if $N[x] = N[y]$, then x and y are adjacent. A graph is *reduced* if any two distinct vertices have distinct closed neighbourhoods.

Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. We call G' a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. If in addition $E' = \{xy \in E|\ x, y \in V'\}$ then we call G' an *induced subgraph* of G. For each $S \subseteq V$, the subgraph of G *induced by* S, denoted by $< S >$ or S, is the unique induced subgraph of G with vertex set S.

Suppose that $G = (V, E)$ is a graph and $S \subseteq V$ is a set of vertices of G. We use $G - S$ to denote the subgraph induced by $V - S$. We write $G - x$ instead of $G - \{x\}$. If S contains no adjacent vertices, then S is called an *independent set* of G.

A graph $G = (V, E)$ is *isomorphic to* a graph $G' = (V', E')$ if there is a one-to-one mapping f from V to V' such that $xy \in E$ if and only if $f(x)f(y) \in E'$. The mapping f is an *isomorphism* from G to G'. If G is isomorphic to G', then we also say that G and G' are *isomorphic*, or G is a copy of G'.

A *walk* of length k in a graph $G = (V, E)$ is a sequence $v_0e_1v_1e_2 \ldots e_kv_k$ where $v_0, v_1, \ldots, v_k$ are vertices, $e_1, e_2, \ldots, e_k$ are edges of G, and v_{i-1} and v_i are the two ends of e_i, $1 \leq i \leq k$. We will call such a walk a (v_0, v_k)-*walk*. If all vertices $v_0, v_1, \ldots, v_k$ and all edges $e_1, e_2, \ldots, e_k$ are distinct, then the walk is a *path*. If $v_0 = v_k$, then the walk is *closed*. A closed walk $v_0e_1v_1e_2 \ldots e_kv_0$ is a *cycle* if $v_i \neq v_j$ and $e_i \neq e_j$ when $i \neq j$. Whenever we deal with graphs without multiple edges, we may suppress the edges and write $P = v_1 \sim v_2 \sim \ldots \sim v_k$ to denote a (v_1, v_k)-walk (resp. (v_1, v_k)-path) and use $C = v_1 \sim v_2 \sim \ldots \sim v_k \sim v_1$ to denote a closed walk (resp. cycle). Vertices v_i and v_{i+1} are called *consecutive vertices.* (The subscript addition is modulo k in the case of C.) A path or a cycle is *chordless* (in a graph G) if non-consecutive vertices are not adjacent (in G). A graph is *chordal* if it contains no chordless cycle of length strictly greater than three.

A graph G is *connected* if there is an (x, y)-path for any two vertices x and y. A *connected component* or simply a *component* of G is a maximal connected subgraph of G. For any two vertices x and y, the length of a shortest (x, y)-path is the *distance* between x and y.

We now define the *substitution operation*: To *substitute* a graph H for a vertex v of a graph G means to form a new graph G' from G by replacing v with H so that in G' every vertex of H is adjacent to every neighbour of v.

Let $\mathcal{F} = \{S_1, S_2, \ldots, S_n\}$ be a family of sets. The *intersection graph* of $\mathcal{F}$ is a graph G with the vertex set $\{v_1, v_2, \ldots, v_n\}$ such that $v_i \sim v_j$ if and only if $S_i \cap S_j \neq \emptyset$. The family $\mathcal{F}$ is called a *representation* of the graph G.

A *circular arc family* is a collection of arcs on a circle. A circular arc family is *inclusion-free* or *proper* if no arc is completely contained in another. A graph G is a *circular arc graph* if it is an intersection graph of a circular arc family; G is a *proper circular arc graph* if it is an intersection graph of a proper circular arc family. An *interval family* is a collection of intervals on the real line. An interval family is *inclusion-free* or *proper* if no interval is completely contained in another. A graph G is an *interval graph* if it is an intersection graph of an interval family; a *proper interval graph* is an intersection graph of a proper interval family. An interval graph is a circular arc graph and a proper interval graph is a proper circular arc graph.

A *digraph* D is an ordered pair (V, A), where V is a finite set and A is a set of ordered pairs xy with $x, y \in V$ (i.e., A is a binary relation on V). The elements of V are *vertices* and the elements of A are *arcs.* In our definition of a digraph, we do not allow multiple arcs, i.e., arcs joining two vertices x and y in the same direction (either all from x to y or all from y to x), and we do not allow loops, i.e., arcs joining a vertex to itself. The vertices x and y are *adjacent* if there is an arc between them. We will use $D = (V, A)$ or simply D to denote a digraph. We use $G(D)$ to denote the underlying graph of D, i.e., the graph with vertex set V and $u \sim v$ if and only if u and v are adjacent. We call $D = (V, A)$ an *oriented graph* if the relation A is antisymmetric.

If xy is an arc of a digraph D, then we say that x *dominates* y or y *is dominated by* x, written as $x \rightarrow y$. We shall write $x \not\rightarrow y$ if x does not dominate y. Suppose that A and B are two subsets of $V(D)$. If every vertex in A is adjacent to every vertex in B, then A and B are *completely adjacent.* If no vertex in A is adjacent to a vertex in B, then A and B are *completely non-adjacent.* If every vertex in A dominates every vertex in B, then we say that A *dominates* B or B *is dominated by* A, and write $A \rightarrow B$.

For any two vertices x and y, if x dominates y, then y is an *out-neighbour* of x, and x is an *in-neighbour* of y. The *outset* of x, denoted by $O(x)$, consists of all out-neighbours of x, and the *closed outset* of x, denoted by $O[x]$, is just $O(x) \cup \{x\}$. The *inset* of x, denoted by $I(x)$, consists of all in-neighbours of x, and the *closed inset* of x, denoted by $I[x]$, is $I(x) \cup \{x\}$. The number of vertices in $O(x)$, denoted by $d^+(x)$, is the *outdegree* of x, and the number of vertices in $I(x)$, denoted by $d^-(x)$, is the *indegree* of x. A digraph D is k*-regular* if all vertices of D have indegree and outdegree k.

Let $D = (V, A)$ and $D' = (V', A')$ be two digraphs. We call D' a *subdigraph* of

D if $V' \subseteq V$ and $A' \subseteq A$. If in addition $A' = \{xy \in A | x, y \in V'\}$ then we call D' an *induced* subdigraph of D. For each $S \subseteq V(D)$, the subdigraph of D *induced by* S, denoted by $< S >$ or S, is the unique induced subdigraph of D with vertex set S. For convenience, we will usually not distinguish a subset S of vertices and the subdigraph induced by S. We define $D - S$ to be the subgraph of D induced by $V - S$. We write $D - x$ instead of $D - \{x\}$.

A *directed path* P of length k is a digraph with the vertex set $\{x_0, x_1, \ldots, x_k\}$ and the arc set $\{x_0 \rightarrow x_1, x_1 \rightarrow x_2, \ldots, x_{k-1} \rightarrow x_k\}$, such that all the vertices and arcs shown are distinct. We will call such a directed path an (x_0, x_k)-path and will denote it by

$$x_0 \rightarrow x_1 \rightarrow x_2 \ldots \rightarrow x_{k-1} \rightarrow x_k.$$

A *directed cycle* C of length k is a digraph with the vertex set $\{v_1, v_2, \ldots, v_k\}$ and the arc set $\{x_1 \rightarrow x_2, x_2 \rightarrow x_3, \ldots, x_{k-1} \rightarrow x_k, x_k \rightarrow x_1\}$. A *hamiltonian path* (resp. *hamiltonian cycle*) in a digraph D is a path (resp. cycle) with the vertex set $V(D)$.

A digraph D is *strong* if for any two vertices x and y there is a directed (x, y)-path and a directed (y, x)-path. A *strong component* of a digraph D is a maximal strong subdigraph. The *strong component digraph* $SC(D)$ of a digraph D is obtained by contracting each strong component to a single vertex (some authors call this the *condensation* of D, [37]).

Suppose that D is a digraph. We define a relation '$\equiv$' on the set $V(D)$: Let x and y be two vertices of D. Then $x \equiv y$ if and only if $N[x] = N[y]$ in $G(D)$. It is easy to see '$\equiv$' is an equivalence relation on $V(D)$. Let $V_1, V_2, \ldots, V_p$ be the equivalence classes of the corresponding partition. We refer to each V_i as a *block*. Then each block induces a semicomplete digraph and two blocks are either completely adjacent or completely non-adjacent.

For digraphs, the *substitution operation* is defined as follows: To *substitute* a digraph S for a vertex v of a digraph D means to form a new digraph D' from D by replacing v with S so that in D' every vertex of S dominates every out-neighbour of v and is dominated by every in-neighbour of v.

A *c-colouring* of a digraph $D = (V, A)$ (or a graph $G = (V, E)$) is a mapping from V to a set C of size c such that two adjacent vertices are mapped to distinct elements. Each element of C is called a *colour*. For each colour, the set of preimages of that colour is called a *colour class*.

A *semicomplete digraph* is a digraph which does not contain non-adjacent vertices. A *tournament* is a semicomplete oriented graph. A *locally semicomplete digraph* is a digraph D in which the outset as well as the inset of each vertex induces a semicomplete digraph, [4]. A *local tournament* is a locally semicomplete digraph which is an oriented graph, [20, 39, 41, 47]. In other words, a local tournament is an oriented graph in which the outset as well as the inset of each vertex induces a tournament.

An oriented graph D is *transitive* if for each vertex x every vertex in $I(x)$ dominates every vertex in $O(x)$. An oriented graph is a *local transitive tournament* is an oriented graph in which the outset as well as the inset of each vertex induces a transitive tournament.

An oriented graph is an *in-tournament* (resp. *out-tournament*) if the inset (resp. the outset) of each vertex induces a tournament, [11]. The class of local tournaments is in fact the intersection of the class of in-tournaments and the class of out-tournaments. An oriented graph D is *locally bicomplete* if for each vertex x every vertex in $I(x)$ is adjacent to every vertex in $O(x)$.

An *orientation* of a graph G is a digraph obtained from G by assigning a direction to every edge of G. In other words, D is an orientation of G if and only if $G = G(D)$.

A graph is *orientable as a local tournament* or *local-tournament-orientable* if there is an orientation D of G which is a local tournament. The oriented graph D is a *local-tournament orientation of G*. Terms such as *locally-transitive-tournament-orientable* (*orientable as a local transitive tournament*), *in-tournament-orientable* (*orientable as an in-tournament*), *local-bicomplete-orientable* (*orientable as a locally bicomplete oriented graph*), and *transitively orientable* (*orientable as a transitive oriented graph*) are defined analogously.

A *full reversal* of a digraph is an operation which reverses the direction of each arc of D. A graph G is *uniquely orientable as a local tournament* if G admits precisely two local-tournament orientations both of which are obtainable from the other by a full reversal.

Chapter 2

Local Tournaments

2.1 Local-tournament Orientability

Recall that a local tournament is an oriented graph in which the outset as well as the inset of each vertex induces a tournament. All tournaments are of course local tournaments. Moreover all directed paths and cycles are also local tournaments. It has been shown in [4] that many nice properties of tournaments are valid for locally semicomplete digraphs and hence also for local tournaments. In particular, a locally semicomplete digraph always has a hamiltonian path, and it has a hamiltonian cycle if and only if it is strong.

We are interested in graphs orientable as local tournaments (recall we refer to them as local-tournament-orientable graphs). Since every induced subgraph of a local tournament is also a local tournament, every induced subgraph of a local-tournament-orientable graph is also local-tournament-orientable.

In 1982, Skrien discovered, in different terminology, the following nice result which gives a full characterization of local-tournament-orientable graphs (cf. [71]). This result was independently found in [49] and we give our proof later in this thesis.

Theorem 2.1.1 *A connected graph is local-tournament-orientable if and only if it is a proper circular arc graph.* □

We emphasize that the condition 'connected' is necessary. For example, a graph consisting of a directed cycle of length 4 and an isolated vertex is local-tournament-orientable but is not a proper circular arc graph. However a general proper circular arc graph is always local-tournament-orientable. Moreover if a graph (not necessarily connected) is local-tournament-orientable, then each connected component must be a proper circular arc graph. Since a proper interval graph is a proper circular arc graph, the following result is an easy consequence of Theorem 2.1.1.

Corollary 2.1.2 *Every proper interval graph is local-tournament-orientable.* □

To determine whether a connected graph is local-tournament-orientable, it is enough, according to Theorem 2.1.1, to verify whether it is a proper circular arc graph. In [81], Tucker gave a matrix characterization of proper circular arc graphs. From it an $O(n^2)$ time algorithm can be obtained to recognize proper circular arc graphs (cf. [61]). In this section, we will give a simple algorithm to recognize local-tournament-orientable graphs. Our algorithm runs in time $O(m\Delta)$, where m is the number of edges and Δ is the maximum degree of the input graph. An optimal algorithm will be given in Chapter 4. In order to describe our algorithm, we give the following notation which is also useful in later discussions.

Let $G = (V, E)$ be a graph. We define

$$F(G) = \{(u, v) |\ uv \in E\}$$

the set of all ordered pairs (u, v) such that uv is an edge of G. Note that each $uv \in E$ gives rise to two ordered pairs (u, v) and (v, u) of $F(G)$. We also define for each subset B of $F(G)$,

$$B^{-1} = \{(u, v) |\ (v, u) \in B\} \text{ and } \hat{\mathrm{B}} = \{uv \in E |\ (u, v) \in B \cup B^{-1}\}.$$

We now define the *characteristic graph* G^* with the vertex set $F(G)$ and adjacency defined by the following: Each $(u, v) \in F(G)$ is adjacent to (v, u), to any $(u, w) \in F(G)$ with $v \neq w$ and $vw \notin E$, to any $(w, v) \in F(G)$ with $u \neq w$ and $uw \notin E$, and to no other vertex of G^*.

Theorem 2.1.3 *A graph G admits a local-tournament orientation if and only if the characteristic graph G^* is 2-colourable.*

Moreover, if G^* is 2-coloured with A being a colour class, then $D = (V, A)$ is a local-tournament orientation of G.

Proof: Suppose that D is a local-tournament orientation of G. We colour the vertices of G^* with two colours μ and τ in the following way: Colour a vertex (u, v) by μ if u dominates v, and by τ if v dominates u in D. Let (x, y) and (x', y') be two adjacent vertices of G^*. It is easy to see that x dominates y if and only if y' dominates x' in D. Hence (x, y) and (x', y') are coloured with different colours. Therefore G^* is 2-colourable.

Suppose now that G^* is 2-coloured with A being a colour class. We prove that $D = (V, A)$ is a local-tournament orientation of G. Since, for each $(u, v) \in F(G)$, (u, v) and (v, u) are adjacent in G^*, exactly one of (u, v) and (v, u) belongs to A. Thus D is an orientation of G. To show that D is a local tournament, let u, v, and w be three vertices of G such that v and w are two non-adjacent neighbours of u. Then $(u, v), (u, w) \in F(G)$ are adjacent in G^* (and (v, u) and (w, u) are adjacent in G^*). Hence at most one of (u, v) and (u, w) (and at most one of (v, u) and (w, u)) belongs to A. Therefore D is a local-tournament orientation of G. □

Theorem 2.1.3 proves the correctness of the following algorithm for recognizing local-tournament-orientable graphs and finding local-tournament orientations.

Algorithm 2.1.4 *Let $G = (V, E)$ be a graph.*

Step 1. Construct the characteristic graph G^* of G.

Step 2. If G^* is not 2-colourable, then G is not local-tournament-orientable.

Step 3. If G^* is 2-colourable, then find any 2-colouring of G^* and obtain a local-tournament orientation $D = (V, A)$ of G where A is a colour class of G^*. □

Theorem 2.1.5 *There is an $O(m\Delta)$ algorithm to recognize local-tournament-orientable graphs and to find such an orientation if there is one.*

Proof: The graph G^* has $O(m)$ vertices, $O(\Sigma_{uv\in E} deg(u) + deg(v)) = O(m\Delta)$ edges and it can be constructed in time $O(m\Delta)$. In the same time we can test, by breadth-first search, whether it is 2-colourable, and find a 2-colouring of G^*. □

Corollary 2.1.6 *There is an $O(m\Delta)$ algorithm to recognize proper circular arc graphs.*

Proof: This is immediate from Theorems 2.1.1 and 2.1.5. □

Let $(u,v),(x,y) \in F(G)$ be two ordered pairs. We say (u,v) *forces* (x,y), denoted by $(u,v)\Gamma(x,y)$, if one of the following conditions is satisfied.

- $u = x$ and $v = y$;
- $v = x$, $u \neq y$, and $uy \notin E$;
- $u = y$, $v \neq x$, and $vx \notin E$.

It is obvious that if $(u,v)\Gamma(x,y)$ then (u,v) is adjacent to (y,x) (and (x,y) is adjacent to (v,u)) in G^*. We say that (u,v) *implies* (x,y), denoted by $(u,v)\Gamma^*(x,y)$, if there exist $(u_1,v_1),(u_2,v_2),\ldots,(u_k,v_k) \in F(G)$ so that

$$(u,v) = (u_1,v_1)\Gamma(u_2,v_2)\Gamma\ldots\Gamma(u_k,v_k) = (x,y).$$

Lemma 2.1.7 *For any graph G, the binary relation Γ^* on $F(G)$ is an equivalence relation.* □

According to Lemma 2.1.7, the equivalence relation Γ^* partitions $F(G)$ into equivalence classes. We call each of these equivalence classes a Γ^**-class*.

Lemma 2.1.8 *Let D be a local-tournament orientation of G. If $(u,v)\Gamma^*(x,y)$ for some (u,v) and (x,y) in $F(G)$, then $u \rightarrow v$ if and only if $x \rightarrow y$ in D.*

Proof: We prove that if $(u,v)\Gamma(x,y)$ for some (u,v), $(x,y) \in F(G)$, then $u \rightarrow v$ if and only if $x \rightarrow y$. The general proof can be done by induction.

If $u = x$ and $v = y$, then the conclusion follows trivially. Suppose that $v = x$, $u \neq y$, and $uy \notin E$. If $u \rightarrow v$ and $y \rightarrow x$ in D, then the vertex v has two non-adjacent in-neighbours, contradicting the fact that D is a local tournament. If $v \rightarrow u$ and $x \rightarrow y$ in D, then the vertex v has two non-adjacent out-neighbours, contradicting the fact that D is a local tournament. A similar proof applies when $u = y$, $v \neq x$, and $vx \notin E$. □

Theorem 2.1.9 *A graph G is orientable as a local tournament if and only if there is no $(u,v) \in F(G)$ such that $(u,v)\Gamma^*(v,u)$.*

Proof: The necessity follows immediately from Lemma 2.1.8. For the sufficiency, assume that there is no $(u,v) \in F(G)$ with $(u,v)\Gamma^*(v,u)$. We apply the following procedure to obtain an orientation D of G. Arbitrarily pick an edge uv which has not been oriented and let $x \rightarrow y$ in D for all (x,y) such that $(u,v)\Gamma^*(x,y)$. Continue the procedure until every edge of G is oriented. Since there is no $(u,v) \in F(G)$ with $(u,v)\Gamma^*(v,u)$, each edge of G is assigned precisely one direction. Thus D is an orientation of G. If D is not a local tournament, then there exists a vertex x such that either $O(x)$ or $I(x)$ is not complete. Assume $O(x)$ is not complete. Then there are two non-adjacent vertices y and z which are dominated by x. Hence $(x,y)\Gamma^*(z,x)$ in $F(G)$ and by the above procedure $x \rightarrow y$ if and only if $z \rightarrow x$ in D. We have $z \rightarrow x$ since $x \rightarrow y$. Therefore we have both $x \rightarrow z$ and $z \rightarrow x$ in D, a contradiction. A similar discussion applies when $I(x)$ is not complete. □

The proof of Theorem 2.1.9 gives an alternative implementation of Algorithm 2.1.4, by working directly on the graph G.

Corollary 2.1.10 *A graph G is orientable as a local tournament if and only if $B \cap B^{-1} = \emptyset$ for any Γ^*-class B.*

Proof: This follows from Lemma 2.1.7, Theorem 2.1.9, and the fact that $B \cap B^{-1} \neq \emptyset$ if and only if B contains both (u,v) and (v,u) for some $(u,v) \in F(G)$. □

Corollary 2.1.11 *If G is a local-tournament-orientable graph with $\Delta(G) = n-1$, then $\overline{G}$ is bipartite.*

Proof: Suppose that $\overline{G}$ is not bipartite. Let $u_1 \sim u_2 \sim \ldots \sim u_{2k+1}$ be an odd cycle in $\overline{G}$. Since $\Delta(G) = n-1$, G contains a vertex v of degree $n-1$. Note that, in G, $v \neq u_i$ and $v \sim u_i$ for each $i = 1, 2, \ldots, 2k+1$. Then

$$(v,u_1)\Gamma(u_2,v)\Gamma \ldots \Gamma(v,u_{2k+1})\Gamma(u_1,v).$$

Hence $(v,u_1)\Gamma^*(u_1,v)$ and by Theorem 2.1.9 G is not local-tournament-orientable, contradicting our hypothesis. □

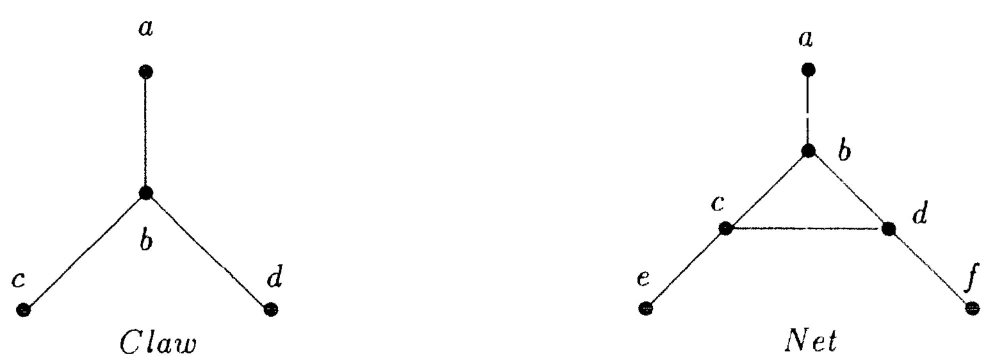

Figure 2.1: The *Claw* and the *Net*

A graph is *claw-free* if it contains no claw (see Fig. 2.1) as an induced subgraph. Similarly a graph is *net-free* if it contains no net as an induced subgraph.

Corollary 2.1.12 *A local-tournament-orientable graph is claw-free and net-free.*

Proof: It suffices to show that neither the claw nor the net is local-tournament-orientable. In the claw, we have

$$(a,b)\Gamma(b,c)\Gamma(d,b)\Gamma(b,a),$$

and in the net, we have

$$(a,b)\Gamma(b,c)\Gamma(c,e)\Gamma(d,c)\Gamma(f,d)\Gamma(d,b)\Gamma(b,a).$$

Therefore, by Theorem 2.1.9, neither the claw nor the net is orientable as a local tournament. □

Proposition 2.1.13 *Let G be a local-tournament-orientable graph and suppose that G^* is coloured with two colours. Then each Γ^*-class consists of all vertices of one colour in one component of G^*.*

Proof: Suppose that A is a Γ^*-class. For any two elements (u,v) and (x,y) of A, by definition of a Γ^*-class, there exist $(x_0,y_0),(x_1,y_1),\ldots,(x_i,y_i)$ such that

$$(u,v)=(x_0,y_0)\Gamma(x_1,y_1)\Gamma\ldots\Gamma(x_i,y_i)=(x,y).$$

Hence, in G^*,

$$(u,v)=(x_0,y_0)\sim(y_0,x_0)\sim(x_1,y_1)\sim\ldots\sim(x_i,y_i)=(x,y)$$

is a path of even length from (u,v) to (x,y). Therefore (u,v) and (x,y) are in the same component and must be coloured with the same colour.

On the other hand, suppose that (u,v) and (x,y) are coloured with the same colour and are in the same component of G^*. Then there is a path of even length from (u,v) to (x,y). Assume that

$$(u,v)=(a_0,b_0)\sim(a_1,b_1)\sim\ldots\sim(a_j,b_j)=(x,y)$$

is such a path. Then

$$(u,v)=(a_0,b_0)\Gamma(b_1,a_1)\Gamma\ldots\Gamma(a_j,b_j)=(x,y).$$

Hence (u,v) and (x,y) are in the same Γ^*-class. □

Suppose that G is local-tournament-orientable and suppose that G^* is coloured with μ and τ. If B is the set of all vertices coloured with μ in one component, then B^{-1} is the set of all vertices coloured with τ in the same component as the vertices of B. Hence both B and B^{-1} are independent in G^*.

Note that by switching the two colours of vertices in a component of G^* we get a new 2-colouring. Therefore if we let

$$F(G) = B_1 \cup B_2 \cup \ldots \cup B_t \cup B_1^{-1} \cup B_2^{-1} \cup \ldots \cup B_t^{-1}$$

be the decomposition of $F(G)$ into Γ^*-classes, then each B_i (and each B_i^{-1}) is an independent set in G^* and each $B_i \cup B_i^{-1}$ induces a component of G^* for each $i = 1, 2, \ldots, t$. Moreover, by Algorithm 2.1.4, a local-tournament orientation of G can be obtained by choosing the arc set to be $A_1 \cup A_2 \cup \ldots \cup A_t$ where $A_i = B_i$ or B_i^{-1} for each $i = 1, 2, \ldots, t$. In fact, it is not difficult to see that this gives all possible local-tournament orientations of G.

Corollary 2.1.14 *A graph G is uniquely local-tournament-orientable if and only if G^* is a connected bipartite graph.* □

We close this section by presenting the following theorem.

Theorem 2.1.15 *The following statements are equivalent for a connected graph G:*

1. *G is a proper circular arc graph;*

2. *G is local-tournament-orientable;*

3. *G is local-transitive-tournament-orientable;*

4. *G^* is 2-colourable;*

5. *$B \cap B^{-1} = \emptyset$ for each Γ^*-class;*

Proof: The equivalence between statements 1 and 2 is Theorem 2.1.1. The equivalence between statements 2 and 4 is basically Theorem 2.1.3. The equivalence between statements 2 and 3 will be proved later (see Corollary 2.2.10). Finally the equivalence between statements 2 and 5 is just Corollary 2.1.10. □

2.2 Local Transitive Tournaments

We call an oriented graph D *straight* [20], if the vertices of D can be linearly ordered $v_1, v_2, \ldots, v_n$ so that each vertex v_i dominates $v_{i+1}, v_{i+2}, \ldots, v_{i+r}$ and is dominated by $v_{i-1}, v_{i-2}, \ldots, v_{i-l}$, where $l = d^-(v_i)$ and $r = d^+(v_i)$. We call such a linear ordering of vertices *a straight enumeration* of D. We say that a graph G is *straight-orientable* if there is an orientation D of G so that D is straight. If G is straight-orientable, then the vertices of G can be linearly ordered $v_1, v_2, \ldots, v_n$ so that for each i there exist $l, r \geq 0$ (which may depend on the subscript i) such that

$$N[v_i] = \{v_{i-l}, \ldots, v_{i-1}, v_i, v_{i+1}, \ldots, v_{i+r}\},$$

where both $\{v_{i-l}, \ldots, v_{i-1}, v_i\}$ and $\{v_i, v_{i+1}, \ldots, v_{i+r}\}$ induce complete subgraphs of G. We also call such a linear ordering of vertices a *straight enumeration* of G. We refer to v_iv_{i-l} as the *left-most wave* at v_i if $l \neq 0$ and to v_iv_{i+r} as the *right-most wave* at v_i if $r \neq 0$.

We call an oriented graph D *round* [20], if the vertices of D can be circularly ordered $v_1, v_2, \ldots, v_n$ so that each vertex v_i dominates $v_{i+1}, v_{i+2}, \ldots, v_{i+r}$ and is dominated by $v_{i-1}, v_{i-2}, \ldots, v_{i-l}$, where $l = d^-(v_i)$, $r = d^+(v_i)$, and subscript additions and subtractions are modulo n. We call such a linear ordering of vertices a *round enumeration* of D. (A round tournament is sometimes called *dominating orientable* cf. [2].) We say that a graph G is *round-orientable* if there is an orientation D of G so that D is round. If G is round-orientable, then the vertices of G can be circularly ordered $v_1, v_2, \ldots, v_n$ so that for each i there exist $l, r \geq 0$ (which may depend on the subscript i) such that

$$N[v_i] = \{v_{i-l}, \ldots, v_{i-1}, v_i, v_{i+1}, \ldots, v_{i+r}\},$$

where both $\{v_{i-l}, \ldots, v_{i-1}, v_i\}$ and $\{v_i, v_{i+1}, \ldots, v_{i+r}\}$ induce complete subgraphs of G and where subscript additions and subtractions here are modulo n. We also call such a circular ordering of vertices *a round enumeration* of G. We refer to $v_i v_{i-l}$ as the *left-most wave* at v_i if $l \neq 0$ and $v_i v_{i+r}$ as the *right-most wave* at v_i if $r \neq 0$.

Suppose that $v_1, v_2, \ldots, v_n$ is a straight enumeration. If, for some $i < j$, $v_i v_j$ is an edge, then $< \{v_i, v_{i+1}, \ldots, v_j\} >$ is complete. Suppose that $v_1, v_2, \ldots, v_n$ is a round enumeration. If $v_i v_j$ is an edge, then at least one of $< \{v_i, v_{i+1}, \ldots, v_j\} >$ and $< \{v_j, v_{j+1}, \ldots, v_i\} >$ is complete. These are useful observations which are frequently employed in the sequel.

We shall see below that the class of connected straight oriented graphs is the same as the class of connected non-strong local transitive tournaments, and the class of connected round oriented graphs is the same as the class of connected local transitive tournaments. First we have the following lemma.

Lemma 2.2.1 *If D is a connected local tournament which contains no directed cycle, then D contains a unique vertex of indegree zero.*

Proof: Since D is acyclic, D contains at least one vertex of indegree zero. On the other hand, if a and b were two distinct vertices of indegree zero, then a and b are non-adjacent, and it is easy to see that the shortest path (which contains no chord) joining a and b in $G(D)$ must contain a vertex with both incident edges oriented towards it, contradicting the fact that D is a local tournament. □

Theorem 2.2.2 *The following four properties are equivalent for a connected oriented graph D:*

1. *D is a non-strong local transitive tournament;*
2. *D is an acyclic local tournament;*
3. *D is a straight oriented graph;*

4. *there exists an inclusion-free family of intervals associated with the vertices of D such that u dominates v in D if and only if the interval associated with u contains the left endpoint of the interval associated with v (the interval of u intersects the interval of v 'on the left').*

Proof: $1 \Rightarrow 2$: Suppose that D is a connected non-strong local transitive tournament. Assume that D contains directed cycles. Let

$$C = x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_l \rightarrow x_1$$

be a longest directed cycle in D. Since D is non-strong, C can not contain all vertices of D. Since D is connected, there exists a vertex y which is not a vertex of C and is adjacent to some vertex x_i of C. Suppose that $x_i \rightarrow y$. (A similar discussion applies if $y \rightarrow x_i$.) Then both x_{i+1} and y are dominated by x_i. Also y and x_{i+1} are adjacent because D is a local tournament. Note that y can not dominate x_{i+1} as otherwise we would obtain a cycle

$$x_1 \rightarrow \ldots \rightarrow x_i \rightarrow y \rightarrow x_{i+1} \rightarrow \ldots \rightarrow x_l \rightarrow x_1$$

of length $l+1$, contradicting the choice of C. Hence x_{i+1} dominates y. Now both x_{i+2} and y are dominated by x_{i+1} and so x_{i+2} is adjacent to y. Again y can not dominate x_{i+2} as otherwise we would obtain a longer cycle in D. So x_{i+2} dominates y. Continuing this discussion, we conclude that each vertex of C dominates y. Therefore C is completely contained in $I(y)$. This is impossible as $I(y)$ must induce a transitive tournament.

$2 \Rightarrow 3$: Suppose that D is an acyclic local tournament. We can obtain a straight enumeration of the vertices of D as follows: Let v_1 be the unique vertex of indegree zero (see Lemma 2.2.1). Assume $v_1, v_2, \ldots, v_k$ have already been defined: let v_{k+1} be the unique out-neighbour of v_k in $D - \{v_1, v_2, \ldots, v_k\}$ which dominates every other out-neighbour of v_k in $D - \{v_1, v_2, \ldots, v_k\}$. (Recall that the outset of v_k is a transitive tournament.) We claim that when v_k has no out-neighbours in $D - \{v_1, v_2, \ldots, v_k\}$, then $k = n$, i.e., all vertices have been ordered. Suppose $k < n$. Since D is connected, there is an edge $v_i w$ or $w v_i$ with w not among $v_1, v_2, \ldots, v_n$. Since the two cases are similar, assume that v_i dominates w. From the definition of v_{i+1} and the fact that D is a local tournament, we see that v_{i+1} must dominate w. Continuing this way we conclude that v_k dominates w, a contradiction. To see that this is a straight enumeration of D, consider a vertex v_i dominating a vertex v_j with $j > i$. Since v_{j-1} always dominates v_j (if $j > 1$), the vertices v_i and v_{j-1} are adjacent. If v_{j-1} dominates v_i, then v_{i-1} and v_{j-1} are adjacent and hence v_{j-1} dominates v_{i-1} (otherwise the choice of v_i was incorrect). Continuing this way we see that v_{j-1} dominates v_1, a contradiction. Therefore v_i must dominate v_{j-1}. It follows that v_i dominates $v_{i+1}, v_{i+2}, \ldots, v_{i+l}$ for some l, and a similar argument shows that it is dominated by $v_{i-1}, v_{i-2}, \ldots, v_{i-k}$ for some k.

$3 \Rightarrow 4$: Given a straight enumeration $v_1, v_2, \ldots v_n$ of D, we associate with v_i the interval on the real line from i to $i + d_i^+ + 1 - \frac{1}{i}$, where d_i^+ is the outdegree of v_i. Then it can be verified that this is a proper interval representation in which the interval u contains the left endpoint of the interval v if and only if in D the vertex associated with u dominates the vertex associated with v.

$4 \Rightarrow 1$: The outset of a vertex x is associated with an inclusion-free family of intervals which all contain the right endpoint of the interval corresponding to x. Thus they are linearly ordered by their left endpoints. Hence the vertices associated with these intervals induce a transitive tournament in D. Since all intervals are linearly ordered on the real line, D is non-strong. Therefore D is a non-strong local transitive tournament. □

Note that the constructions in the above proof can all be performed in time $O(m+n)$. In fact, given a non-strong local transitive tournament, it takes $O(m+n)$ time to construct a straight enumeration. Now given a straight enumeration, it takes $O(n)$ time to construct a proper circular arc representation.

Corollary 2.2.3 *Suppose an acyclic local-tournament orientation D of a proper interval graph G is given. Then a straight enumeration D, and hence an interval representation G, can be found in time $O(m+n)$.* □

The following lemma is taken from [4].

Lemma 2.2.4 *Suppose that D is a connected non-strong local tournament. Then the strong components of D can be linearly ordered $C_1, C_2, \ldots, C_k$ so that*

1. *each C_i is complete, $i = 1, 2, \ldots, k$,*
2. *$C_i \rightarrow C_{i+1}$, $i = 1, 2, \ldots, k-1$,*
3. *if $j < i$ then no vertex in C_i dominates a vertex in C_j,*
4. *if $i \neq j$ then C_i and C_j are completely adjacent or completely non-adjacent, and*
5. *if $C_i \rightarrow C_j$ then $C_i \cup C_{i+1} \ldots \cup C_j$ is complete.* □

Theorem 2.2.5 *If a connected graph G admits a non-strong local-tournament orientation, then G admits a straight enumeration.*

Proof: Suppose that D is a non-strong local-tournament orientation of G. Then the strong components of D can be linearly ordered $C_1, C_2, \ldots, C_k$ so that the properties in Lemma 2.2.4 hold. We form an oriented graph D' from D in the following way: Replace each C_i by a transitive tournament T_i of order $|V(C_i)|$. Note that $T_i \rightarrow T_j$ in D' if and only if $C_i \rightarrow C_j$ in D. We see that D' is an orientation of G. Since T_i is a transitive tournament, the vertices of T_i can be linearly ordered $c_{i,1}, c_{i,2}, \ldots, c_{i,l_i}$ so that $c_{i,p} \rightarrow c_{i,q}$ if $p < q$. We prove that the following linear order of the vertices is a straight enumeration of D':

$$c_{1,1}, \ldots, c_{1,l_1}, c_{2,1}, \ldots, c_{2,l_2}, \ldots, c_{k,1}, \ldots, c_{k,l_k}.$$

Consider a vertex $c_{i,p}$. By property 3 of Lemma 2.2.4, $c_{i,p} \not\rightarrow c_{j,q}$ if $j < i$ or $i = j$ and $q < p$. We know that $c_{i,p}$ dominates $c_{i,(p+1)}, c_{i,(p+2)}, \ldots, c_{i,l_i}$. Furthermore, if $c_{i,p}$ dominates $c_{j,q}$ for some $j > i$, then $c_{i,p}$ also dominates $c_{(i+1),1}, \ldots, c_{(i+1),l_{i+1}}, \ldots, c_{j,1}, \ldots, c_{j,l_j}$ according to properties 3, 4, and 5 of Lemma 2.2.4. Hence the out-neighbours of $c_{i,p}$ appear consecutively succeeding $c_{i,p}$. A similar argument applies to show that the in-neighbours of $c_{i,p}$ appear consecutively preceding $c_{i,p}$. □

Let i be a circular arc on a circle. Suppose that i begins at p and ends at q in the clockwise direction of the circle. We call p the *head* and q the *tail* of i, denoted by $h(i)$ and $t(i)$, respectively.

Theorem 2.2.6 *The following three properties are equivalent for a connected oriented graph D:*

1. *D is a local transitive tournament;*

2. *D is a round oriented graph;*

3. *there exists an inclusion-free family of circular arcs associated with the vertices of D such that $u \rightarrow v$ in D if and only if the circular arc associated with u contains the head of the circular arc associated with v.*

Proof: We only need to show that the properties are equivalent for the case when D is strong, as for the non-strong case we have Theorem 2.2.2. We remark that in the entire proof the subscript additions and subtractions are modulo n.

$1 \Rightarrow 2$: Suppose that D is a connected local transitive tournament. We obtain a round enumeration of D as follows: Start with any vertex v_1. Assume that $v_1, v_2, \ldots, v_k$ have already been defined. Let v_{k+1} be the unique out-neighbour of v_k in $D - \{v_1, v_2, \ldots, v_k\}$ which dominates every other out-neighbour of v_k in $D - \{v_1, v_2, \ldots, v_k\}$. We claim that when v_k has no out-neighbours in $D - \{v_1, v_2, \ldots, v_k\}$, then $k = n$, i.e., all vertices have been ordered. For suppose $k < n$. Since D is connected and D is strong, there is an arc $v_i w$ with w not among $v_1, v_2, \ldots, v_n$. From the definition of v_{i+1} and the fact that D is a local tournament, we see that v_{i+1} must dominate w. Continuing this way we conclude that v_k dominates w, a contradiction.

To prove that the above vertex ordering is a round enumeration, it suffices to show that if $v_i \rightarrow v_j$ then $v_i \rightarrow \{v_{i+1}, v_{i+2}, \ldots, v_{j-1}\}$ and $\{v_i, v_{i+1}, \ldots, v_{j-1}\} \rightarrow v_j$. First we notice that

$$v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_n.$$

We claim that $v_n \rightarrow v_1$. Since D is strong, v_n must dominate some vertex v_a. If $a = 1$, then we are done. Otherwise v_a is dominated by both v_{a-1} and v_n. Since D is a local tournament, either $v_{a-1} \rightarrow v_n$ or $v_n \rightarrow v_{a-1}$. However v_{a-1} cannot dominate v_n by the choice of v_a. Hence v_n dominates v_{a-1}. Continuing this way, we conclude that v_n dominates v_1. So if $j = i + 1$, then we are done. If $j \neq i + 1$, then both v_{i+1} and v_j are dominated by v_i and so, by the choice of v_{i+1}, v_{i+1} dominates v_j. Continuing this way we see that v_j is dominated by each v_m where $m = i, i + 1, \ldots, j - 1$. Since D is a local transitive tournament, we know that $\{v_i, v_{i+1}, \ldots, v_{j-1}\}$ induces a transitive tournament. Since $v_i \rightarrow v_{i+1} \rightarrow \ldots \rightarrow v_{j-1}$, we have $v_i \rightarrow \{v_{i+1}, v_{i+2}, \ldots, v_{j-1}\}$.

$2 \Rightarrow 3$: Suppose that $v_1, v_2, \ldots, v_n$ is a round enumeration of D. Make a n-scale-clock on a cycle. We associate with each vertex v_j a circular arc from j to $(j + d_j^+ + 1) - \frac{1}{j}$ (here additions are modulo n), where d_j^+ is the outdegree of vertex v_j. It is not difficult to verify that this is a proper circular arc representation in which the circular arc u contains the head of the circular arc v if and only if in D the vertex associated with u dominates the vertex associated with v.

$3 \Rightarrow 1$: The outset of a vertex x is associated with an inclusion-free family of circular arcs which all contain the tail of the circular arc associated with x. Thus they are linearly ordered by their heads. Hence the vertices associated with these circular arcs induce a transitive tournament in D. A similar discussion applies to the inset of x, and hence D is a local transitive tournament. □

Again there are two procedures involved in the above proof. One is to obtain a round enumeration from a local transitive tournament, the other is to obtain a proper circular arc representation. The first procedure can be performed in time $O(m + n)$ and the second one can be performed in time $O(n)$.

Corollary 2.2.7 *Suppose a local-transitive-tournament orientation D of a proper circular arc graph G is given. Then a round enumeration of D, and hence a circular arc representation of G, can be found in time $O(m + n)$.* □

The following lemma due to Golumbic can be found in [33].

Lemma 2.2.8 *If G is a proper circular arc graph, then G has a proper circular arc representation in which no two arcs share a common endpoint and no two arcs together cover the entire circle.* □

Theorem 2.2.9 *If a connected graph G is orientable as a local tournament, then G admits a round enumeration.*

Proof: Suppose that G is orientable as a local tournament. Then by Theorem 2.1.1, G has a proper circular arc representation $\mathcal{F}$. In addition, by Lemma 2.2.8 the representation $\mathcal{F}$ can be chosen so that no two arcs share a common endpoint and no two arcs together cover the entire circle. Let $S_1, S_2, \ldots, S_n$ be the arcs in $\mathcal{F}$. For each $i = 1, 2, \ldots, n$, suppose that v_i is the vertex of G associated with S_i.

We obtain an oriented graph D as follows: The vertex set of D is the same as the vertex set of G, and a vertex v_i dominates a vertex v_j in D if and only if S_i contains the head of S_j (or S_j contains the tail of S_i). By the assumption on $\mathcal{F}$, if v_i and v_j are two adjacent vertices of G, then either S_i contains the head of S_j or S_j contains the head of S_i. Thus D is an orientation of G. We claim that D is a local transitive tournament. In fact for each vertex v_i the outset of v_i consists of the vertices which are associated with those arcs containing the tail of S_i and hence they intersect each other. So the outset of v_i induces a complete subgraph of D. By the assumption on $\mathcal{F}$, if an arc contains the tail of S_i, then it can not contain the head of S_i. Hence the arcs which contain the tail of S_i cannot cover the whole circle. Thus the subgraph induced by the outset of v_i can not contain any cycle. A similar argument can be applied to show that the inset of v_i induces a complete subgraph which contains no cycle. Hence D is a local-transitive-tournament orientation of G. Therefore by Theorem 2.2.6 G admits a round orientation. □

From the proof of Proposition 2.2.9, we see that a local-tournament-orientable graph is in fact local-transitive-tournament-orientable.

Corollary 2.2.10 *A connected graph is local-tournament-orientable if and only if it is local-transitive-tournament-orientable.* □

2.3 Two Structure Theorems

Suppose that G is local-tournament-orientable. Let B be a Γ^*-class and let $C = B^{-1}$. Then from Section 2.1 we know that $B \cap C = \emptyset$ and $\hat{B} = \hat{C}$. We call $\hat{B}$ an *implication class* of G. Note that the edge set of G can be partitioned into implication classes. Suppose that $uv, u'v' \in E$ are in the same implication class. Consider (u,v) and (u',v') in $F(G)$. Then either both (u,v) and (u',v') are in B for some Γ^*-class B, or (u,v) is in B and (u',v') is in B^{-1}. Hence either $(u,v)\Gamma^*(u',v')$ or $(u,v)\Gamma^*(v',u')$.

From the above discussion, we see that G has a unique local-tournament orientation if and only if the edge set of G forms one implication class. One observation is that if u and v are two vertices of G with $N[u] = N[v]$, then u is adjacent to v and the single edge uv forms an implication class.

An edge xy of a graph G is called *balanced* if $N[x] = N[y]$ and *unbalanced* if $N[x] \neq N[y]$. Similarly an arc xy of an oriented graph D is called *balanced* if $N[x] = N[y]$ and *unbalanced* if $N[x] \neq N[y]$ in $G(D)$. (Thus a balanced arc joins two '$\equiv$'-equivalent vertices.)

We defined the full reversal of a digraph D to be the operation which reverses the direction of each arc of D. We now define a *partial reversal* of a digraph D to be an operation which reverses the directions of all unbalanced arcs within one component of $\overline{G(D)}$, or reverses directions of all unbalanced arcs between two fixed components of $\overline{G(D)}$. Note that in a digraph D we can perform several different partial reversals. In the remainder of this chapter we shall prove the following two structure theorems.

Theorem 2.3.1 *Let D be a connected oriented graph which is not a tournament. Then D is a non-strong local tournament if and only if it is obtained from some straight oriented graph S with reduced $G(S)$ and $|S| > 1$ by substituting a tournament T_v for each vertex $v \in V(S)$.*

Moreover every non-strong local-tournament orientation of $G(D)$ is obtained from D by reversing the directions of some balanced arcs, possibly followed by a full reversal. □

Theorem 2.3.2 *Let D be a connected oriented graph. Then D is a local tournament if and only if it is obtained from some round oriented graph R with reduced $G(R)$ first by substituting a tournament T_v for each vertex $v \in V(R)$ and then by performing partial reversals.*

Moreover, every local-tournament orientation of $G(D)$ is obtained by performing partial reversals and changing directions of some balanced arcs, possibly followed by a full reversal. □

In what follows the notation $N[x]$ always refers to the closed neighbourhood of x in the graph G. We begin with the following proposition.

Proposition 2.3.3 *Let G be a reduced connected graph with $\Delta(G) \leq n-2$. If G is orientable as a non-strong local tournament, then G has exactly one implication class.*

Proof: Since G can be oriented as a non-strong local tournament, G admits a straight enumeration by Theorem 2.2.5. Let $v_1, v_2, \ldots, v_n$ be a straight enumeration of G. Note that $n > 3$ as G is connected with $\Delta(G) \leq n-2$. For each vertex v_i, v_i is not adjacent to either v_1 or v_n; otherwise v_i would be adjacent to every other vertex of G, contradicting the hypothesis that $\Delta(G) \leq n-2$. Since (by the same argument) v_1 is not adjacent to v_n, we see that $\overline{G}$ is connected.

Fix a vertex v_i where $1 < i < n$. Let v_iv_j be the left-most wave and let v_iv_k be the right-most wave at v_i. We prove that v_iv_j and v_iv_k are in the same implication class. If v_j is not adjacent to v_k, then $(v_i, v_j)\Gamma(v_k, v_i)$ and so v_iv_j and v_iv_k are in the same implication class. Suppose that v_j is adjacent to v_k. Since G is reduced, we have that $N[v_i] \neq N[v_j]$ and $N[v_i] \neq N[v_k]$. Note that $N[v_i] \subset N[v_j]$ and $N[v_i] \subset N[v_k]$. Then $N[v_j] - N[v_i] \neq \emptyset$ and $N[v_k] - N[v_i] \neq \emptyset$. Let v_jv_l be the left-most wave at v_j and

$v_k v_m$ be the right-most wave at v_k. Then $l < j < i < k < m$, and neither v_l nor v_m is adjacent to v_i. Hence

$$(v_i, v_j)\Gamma(v_j, v_l)\Gamma(v_k, v_j)\Gamma(v_m, v_k)\Gamma(v_k, v_i)$$

and so $v_i v_j$ and $v_i v_k$ are in the same implication class.

We now show that all left-most waves and all right-most waves at all vertices are in one implication class. Let $v_1 = v_{i_1} \sim v_{i_2} \sim \ldots \sim v_{i_r} = v_n$ $(i_1 < i_2 < \ldots < i_r)$ be a path of G such that $v_{i_j} v_{i_{j+1}}$ is the right-most wave at v_{i_j} for each $j = 1, 2, \ldots, r-1$. We first notice that $r \geq 4$ as otherwise there is a vertex of degree $n-1$, contrary to our hypothesis. Since v_{i_j} is not adjacent to $v_{i_{j+2}}$ for each $j = 1, 2, \ldots, r-2$, we have that $(v_{i_1}, v_{i_2})\Gamma(v_{i_2}, v_{i_3})\Gamma \ldots \Gamma(v_{i_{r-1}}, v_{i_r})$. Hence the edges $v_{i_j} v_{i_{j+1}}$ where $j = 1, 2, \ldots, r-1$ are in the same implication class. We denote this implication class by $\mathcal{C}$ and claim that all right-most waves at all vertices are in $\mathcal{C}$ and hence also all left-most waves at all vertices are in $\mathcal{C}$. Let $v_s v_t$ be the right-most wave at v_s. Suppose first that v_t is the last vertex in the straight enumeration, namely, $t = n$. Then $s > i_{r-2}$ as otherwise the edge $v_s v_n$ implies that $v_{i_{r-2}}$ is adjacent to v_n, contradicting the fact that $v_{i_{r-2}} v_{i_{r-1}}$ is the right-most wave at $v_{i_{r-2}}$. If in addition $s \leq i_{r-1}$, then $(v_n, v_s)\Gamma(v_s, v_{i_{r-2}})\Gamma(v_{i_{r-2}}, v_{i_{r-3}})$ and so $v_s v_n$ and $v_{i_{r-2}} v_{i_{r-3}}$ are in the same implication class, namely, $\mathcal{C}$. If $i_{r-1} < s < n$, let $v_s v_p$ the left-most wave at v_s, then $i_{r-2} < p$ and $v_p \in N[v_s] - N[v_n]$. Thus v_p is not adjacent to v_n. Hence $(v_n, v_s)\Gamma(v_s, v_p)\Gamma(v_p, v_{i_{r-2}})\Gamma(v_{i_{r-2}}, v_{i_{r-3}})$ and so $v_s v_n$ is in $\mathcal{C}$. Suppose that v_t is not the last vertex in the straight enumeration, namely, $t < n$. Let $v_t = v_{t_1}, v_{t_2}, \ldots, v_{t_q} = v_n$ be a sequence of vertices such that $t_1 < t_2 < \ldots < t_q$ and $v_{t_j} v_{t_{j+1}}$ is the right-most wave at v_{t_j} for each $j = 1, 2, \ldots, q-1$. We note that $v_s v_t$ is in the same implication class as $v_{t_1} v_{t_2}$ and $v_{t_j} v_{t_{j+1}}$ is in the same implication class as $v_{t_{j+1}} v_{t_{j+2}}$ for each $j = 1, 2, \ldots, q-2$. Then $v_s v_t$ is in the same implication class as $v_{t_{q-1}} v_{t_q} = v_{t_{q-1}} v_n$ which is in $\mathcal{C}$. Hence $v_s v_t$ is also in $\mathcal{C}$.

Finally suppose that $v_i v_j$ is any edge of G where $i < j$. Since $N[v_i] \neq N[v_j]$, either $N[v_i] - N[v_j] \neq \emptyset$ or $N[v_j] - N[v_i] \neq \emptyset$. If $N[v_i] - N[v_j] \neq \emptyset$, then $v_i v_j$ is in the same implication class as the left-most wave at v_i which is in $\mathcal{C}$. If $N[v_j] - N[v_i] \neq \emptyset$, then $v_i v_j$ is in the same implication class as the right-most wave at v_i which is is $\mathcal{C}$. Therefore $v_i v_j$ is in $\mathcal{C}$. ☐

Suppose that G is a reduced connected graph with $\Delta(G) \leq n-2$. If G is orientable as a ncn-strong local tournament, then by Proposition 2.3.3 G is uniquely local-tournament-orientable. If $v_1, v_2, \ldots, v_n$ is a straight enumeration of G, then a non-strong local-tournament orientation of G can be obtained by letting $v_i \rightarrow v_j$ for any edge v_iv_j of G with $i < j$. Therefore the following corollary has been proved.

Corollary 2.3.4 *Let G be a reduced connected graph with $\Delta(G) \leq n-2$. If G is orientable as a non-strong local tournament, then G is uniquely orientable as a non-strong local tournament.* □

Proposition 2.3.5 *Let $G = (V, E)$ be a reduced connected graph with $|V| > 3$ and assume that* $\deg(v) = n - 1$ *for some $v \in V$. If G can be oriented as a non-strong local tournament, then G has precisely two implication classes: One class consists of all edges that are incident with v, the other class consists of all edges that are not incident with v.*

Proof: Since $G = (V, E)$ is orientable as a non-strong local tournament, by Theorem 2.2.5, G admits a straight enumeration. Let $v_1, v_2, \ldots, v_n$ be a straight enumeration of G. Since G is reduced and $deg(v) = n-1$, we know that $deg(u) < n-1$ for all $u \neq v$. Let $v = v_r$. We claim that n is odd, $r = \frac{n+1}{2}$, and v_iv_{i+r-1} is the right-most wave at v_i for each $1 \leq i \leq r$.

We first apply induction on i to show that v_iv_{i+r-1} is the right-most wave at v_i for each $1 \leq i \leq r$. Since v_r is adjacent to every other vertex, $v_1v_r \in E$. If $v_1v_j \in E$ for some $j > r$, then v_j is adjacent to $v_1, v_2, \ldots, v_{j-1}$. Since v_r is adjacent to v_n, v_j is adjacent to $v_{j+1}, v_{j+2}, \ldots, v_n$. Thus $deg(v_j) = n-1$, contradicting the fact that v_r is the only vertex of degree $n-1$. Hence $v_1v_j \notin E$ for any $j > r$ and so v_1v_r is the right-most wave at v_1. Suppose that v_lv_{l+r-1} is the right-most wave at v_l for all $l < i \leq r$. Consider the vertex v_i. Since G is reduced, $N[v_{i-1}] \neq N[v_i]$ and $N[v_{i-1}] \subset N[v_i]$, $N[v_i] - N[v_{i-1}] \neq \emptyset$. Hence there is a vertex which is adjacent to v_i but not to v_{i-1}. We claim that $N[v_i] - N[v_{i-1}] = \{v_{i+r-1}\}$. In fact, let $v_j \in N[v_i] - N[v_{i-1}]$. We know that $j > i + r - 2$ because $v_{i-1}v_{i+r-2}$ is the right-most wave at v_{i-1}. If $j > i + r - 1$, then vertices v_j and v_{i+r-1} would have the same closed neighbourhood, contradicting the hypothesis that G is reduced. Hence $j = i + r - 1$ and so v_{i+r-1} is the only vertex which is adjacent to v_i but not v_{i-1}. This also implies that v_iv_{i+r-1} is the right-most wave at v_i. Hence v_iv_{i+r-1} is the right-most wave at v_i for all $1 \leq i \leq r$. In particular, we have v_rv_{2r-1} is the right-most wave at v_r. Therefore $n = 2r - 1$ which is odd and so $r = \frac{n+1}{2}$.

The vertex v_1 is not adjacent to v_n. Each vertex v_i is not adjacent to v_n when $i < r$ and is not adjacent to v_1 when $i > r$. Thus $\overline{G}$ has precisely two components induced by $\{v_r\}$ and $V - \{v_r\}$. First we note that an edge of G which is incident with v_r can not be in the same implication class as an edge which is not incident with v_r. Assume now that v_iv_r and v_jv_r are two edges of G which are incident with v_r. Then v_i and v_j are two vertices in the set $V - \{v_r\}$ which induces a connected subgraph in $\overline{G}$. Hence there is a path in $\overline{G}$ from v_i to v_j. Let $v_i = x_1 \sim x_2 \sim \ldots \sim x_l = v_j$ be such a path. Since, for each $t = 1, 2, \ldots, l - 1$, $(x_t, v_r)\Gamma(v_r, x_{t+1})$, x_tv_r and $x_{t+1}v_r$ are in the same implication class of G. Hence v_iv_r and v_jv_r are in the same implication class.

Let G' be the graph obtained from G by removing the vertex v_r. Then $\Delta(G') \leq n - 2$. Moreover G' is connected (as $n > 3$) and is straight-orientable. Hence by Proposition 2.3.3, G' has only one implication class. Therefore the set of all edges of G which are not incident with the vertex v_r form one implication class. □

We remark that if G is a reduced connected non-strong local-tournament-orientable graph with $\Delta(G) = n - 1$, then G contains at least 3 vertices. When G contains exactly 3 vertices, G is a path of length 2. In this case, G has one implication class and G admits a unique local-tournament orientation. In fact, the orientation must be non-strong. If G has at least 5 vertices (we know from the proof of Proposition 2.3.5 that G must have an odd number of vertices), then G admits precisely two local-tournament orientations up to full reversal. One orientation D can be obtained by letting $v_i \rightarrow v_j$ if v_iv_j is an edge of G with $i < j$ in the straight enumeration. Of course this is a non-strong orientation. Another local-tournament orientation can be obtained from D by reversing all directions of the arcs incident with v_r (the vertex of degree $n-1$). This is a strong orientation. Hence we conclude from Proposition 2.3.5 that there are (up to full reversal) no other local-tournament orientations of G. Hence G is uniquely orientable as a non-strong local tournament. Therefore the following corollary has been proved.

Corollary 2.3.6 *Let $G=(V,E)$ be a reduced connected graph with $\Delta(G)=n-1$. If G can be oriented as a non-strong local tournament, then G is uniquely orientable as a non-strong local tournament.* □

The following result is a combination of Corollaries 2.3.4 and 2.3.6.

Corollary 2.3.7 *Let $G=(V,E)$ be a reduced connected graph. If G is orientable as a non-strong local tournament, then G is uniquely orientable as a non-strong local tournament.* □

Proof of Theorem 2.3.1: Let D be a connected oriented graph which is not a tournament. Suppose that D is obtained from some straight oriented graph S with reduced $G(S)$ by substituting a tournament T_v for each vertex v of S. Let $x_1, x_2, \ldots, x_l$ be a straight enumeration of S. We know that $l \neq 1$ since D is not a tournament. It is implied by the definition of a straight enumeration that there is no directed path from x_l to x_1 in S. Then there is no directed path from any vertex of T_{x_l} to any vertex of T_{x_1} in D. Hence D is non-strong. To see that D is a local tournament, let x be a vertex of D and let y and z be two out-neighbours of x. Then $x \in T_{x_i}$, $y \in T_{x_j}$, and $z \in T_{x_k}$ for some $1 \leq i,j,k \leq l$. We note that $i \leq j,k$. Without loss of generality, assume that $j \leq k$. If $j=k$, then y and z are in T_{x_j} which is a tournament. Hence y and z are adjacent. Assume that $j<k$. Since $x \rightarrow z$, we have that $x_i \rightarrow x_k$ and hence $x_r \rightarrow x_k$ for every r such that $i \leq r \leq k$. In particular, $x_j \rightarrow x_k$. So $y \rightarrow z$ and y is adjacent to z in D. A similar argument applies to show that any two in-neighbours of x are adjacent. Hence D is a local tournament.

Suppose now that D is a non-strong local tournament. Let $T_1 \cup T_2 \cup \ldots \cup T_p$ be a partition of D into blocks. Then each T_i must be a tournament and if $i \neq j$ then T_i is either completely adjacent to T_j or completely non-adjacent to T_j. Since D is not a tournament, we have $p \neq 1$. We note that $p \neq 2$ as otherwise T_1 and T_2 are completely adjacent. Thus vertices in $T_1 \cup T_2$ have the same closed neighbourhood, contradicting the maximality of T_1. Therefore $p \geq 3$.

Let $x_1, x_2, \ldots, x_p$ be vertices with $x_i \in T_i$ $(i = 1, 2, \ldots, p)$ such that $\{x_1, x_2, \ldots, x_p\}$ induces a non-strong subgraph of D. We use S to denote the subgraph of D induced by $\{x_1, x_2, \ldots, x_p\}$. Note that such vertices $x_1, x_2, \ldots, x_p$ must exist because D is non-strong. Then S is a local tournament. We note that $G(S)$ is reduced because distinct vertices of S have distinct closed neighbourhoods. By Theorem 2.2.5, S admits a straight enumeration. Without loss of generality, assume that $x_1, x_2, \ldots, x_p$ is a straight enumeration of S.

Let $\{y_1, y_2, \ldots, y_p\}$ be an arbitrary set of vertices with $y_i \in T_i$ for each $i = 1, 2, \ldots, p$ (possibly the same as $\{x_1, x_2, \ldots, x_p\}$). Suppose that S' is the subgraph of D induced by $\{y_1, y_2, \ldots, y_p\}$. Then S' is also a local tournament with reduced $G(S')$. It is easy to verify that the mapping $f: x_i \rightarrow y_i$ is an isomorphism between $G(S)$ and $G(S')$. By Corollary 2.3.7, $G(S)$ is uniquely orientable as a non-strong local tournament. Hence, under the same isomorphism f, S is either isomorphic to S' or the full reversal of S'. Thus either $y_1, y_2, \ldots, y_p$ or $y_p, y_{p-1}, \ldots, y_1$ is a straight enumeration of S'.

We first consider the case when $p = 3$. In this case S is a directed path $x_1 \rightarrow x_2 \rightarrow x_3$ where x_1 and x_3 are not adjacent. For any $y \in T_1$, we must have $y \rightarrow x_2 \rightarrow x_3$. Hence $T_1 \rightarrow x_2$. Similarly we must have $x_2 \rightarrow T_3$. For any $z \in T_2$, by considering the set $\{x_1, z, x_3\}$, we must have either $x_1 \rightarrow z \rightarrow x_3$ or $x_3 \rightarrow z \rightarrow x_1$. Hence either $T_1 \rightarrow z \rightarrow T_3$ or $T_3 \rightarrow z \rightarrow T_1$.

Let $H_1 \cup H_2$ be a partition of T_2 such that each vertex $y \in H_1$ satisfies $T_1 \rightarrow y \rightarrow T_3$ and each vertex $z \in H_2$ satisfies that $T_3 \rightarrow z \rightarrow T_1$. Then $H_1 \neq \emptyset$ as $x_2 \in H_1$. If $H_2 \neq \emptyset$, then we have $T_1 \rightarrow H_1 \rightarrow T_3 \rightarrow H_2 \rightarrow T_1$, which contradicts the fact that D is non-strong. Hence $H_2 = \emptyset$, that is, $T_1 \rightarrow T_2 \rightarrow T_3$. Thus D is obtained from S by substituting a tournament T_i for x_i for each $i = 1, 2, 3$.

To prove the second assertion of Theorem 2.3.1 for the case when $p = 3$, suppose that D' is any non-strong local tournament with $G(D') = G(D)$. Then D' can be partitioned into vertex disjoint subgraphs $T_1' \cup T_2' \cup T_3'$, where $G(T_i') = G(T_i)$ for each $i = 1, 2, 3$. Again we must have either $T_1' \rightarrow T_2' \rightarrow T_3'$ or $T_3' \rightarrow T_2' \rightarrow T_1'$. If $T_1' \rightarrow T_2' \rightarrow T_3'$, then D' is obtained from D by reversing some arcs in T_i for each $i = 1, 2, 3$. If $T_3' \rightarrow T_2' \rightarrow T_1'$, then D' is obtained from D by first reversing some arcs in T_i and then performing a full reversal. (Note that each arc of T_i is balanced.)

Assume now that $p \geq 4$. Let y be any vertex of T_i and let S'' be the subgraph induced by $\{x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_p\}$. Applying an argument similar to the above, we have either $x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_p$ or $x_p, \ldots, x_{i+1}, y, x_{i-1}, \ldots, x_1$ is a straight enumeration of S''. However S and S'' have at least one arc (x_j, x_{j+1}) in common for some j. Then $x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_p$ must be a straight enumeration of S''. Moreover $y \rightarrow x_j$ if and only if $x_i \rightarrow x_j$, and $x_j \rightarrow y$ if and only $x_j \rightarrow x_i$. This implies that $T_i \rightarrow T_j$ if and only if $x_i \rightarrow x_j$. Thus D is obtained from S by substituting T_i for x_i for each $i = 1, 2, \ldots, p$.

The second assertion of Theorem 2.3.1 for the case $p \geq 4$ can be proved in the same way as in the case when $p = 3$. □

For the case when D is a non-strong tournament, D can be viewed as an oriented graph obtained from a straight oriented graph which has only one vertex x by replacing x with D. In this case each edge of $G(D)$ forms an implication class. Hence all non-strong local-tournament orientations of $G(D)$ are obtained from D by reversing some arcs in D.

Proposition 2.3.8 *Let G be a reduced connected graph. If G is local-tournament-orientable and $\overline{G}$ is bipartite, then the edges of G within each fixed connected component of $\overline{G}$ form one implication class, and the edges of G between any two fixed connected components of $\overline{G}$ form an implication class.*

Proof: Suppose that G is orientable as a local tournament and that $\overline{G}$ is bipartite. Let $G_1 \cup G_2 \cup \ldots \cup G_r$ be a decomposition of G into vertex disjoint subgraphs such that each $\overline{G_i}$ is a connected component of $\overline{G}$. Then, in G, every vertex of G_i is adjacent to every vertex of G_j if $i \neq j$. Hence any implication class is contained in $E(G_i)$ for some i or is contained in the set of all edges between G_i and G_j for some fixed i and j. It suffices to show that all edges of G in any G_i or between any two fixed G_i and G_j are in the same implication class.

First we show that all edges of G within one G_i are in the same implication class. Note that $\overline{G_i}$ is a connected bipartite graph. Let (S, H) be a bipartition of $\overline{G_i}$. Note that S and H induce complete subgraphs in G. We begin by showing that all edges of G within S are in the same implication class. Similarly we can show that all edges of G within H are in the same implication class. If $1 \leq |S| \leq 2$, then S contains no edge or contains exactly one edge and so the statement is trivially true. If $|S| \geq 3$, it suffices to show that for three vertices x, y, z of S, the edges xy and xz are in the same implication class. Since $\overline{G_i}$ is connected, there is a path in $\overline{G_i}$ from y to z. Let $y = w_1 \sim w_2 \sim \ldots \sim w_t = z$ be any shortest (y, z)-path. Note that vertices w_i are taken from S and H alternatively, thus t is odd. It will be enough to prove (for each odd i) that if $w_i \neq x$, then xw_i and xw_{i+2} are in the same implication class, unless $w_{i+2} = x$ in which case xw_i and xw_{i+4} are the same implication class. Assume that $w_{i+2} = x$. Note that $w_{i+1}w_{i+4}$ and w_iw_{i+3} must be edges of G, as otherwise we would have a shorter path in $\overline{G}$ from w_i to w_{i+4} and a shorter (y, z)-path, contradicting our choice of the (y, z)-path. Hence we have

$$(x, w_i)\Gamma(w_i, w_{i+3})\Gamma(w_{i+3}, w_{i+1})\Gamma(w_{i+1}, w_{i+4})\Gamma(w_{i+4}, x).$$

Assume next that $w_{i+2} \neq x$. If xw_{i+1}, then

$$(x, w_i)\Gamma(w_{i+1}, x)\Gamma(x, w_{i+2}).$$

Otherwise we consider the following four cases, one of which must occur because G is reduced.

Case 1. Assume that there exist vertices $v \in N[w_i] - N[x]$ and $u \in N[w_{i+2}] - N[x]$. Then $u, v \in H$. Thus

$$(x, w_i)\Gamma(w_i, v)\Gamma(v, w_{i+1})\Gamma(w_{i+2}, v)\Gamma(x, w_{i+2})$$

if vw_{i+2} is an edge in G; or

$$(x, w_i)\Gamma(w_i, u)\Gamma(u, w_{i+1})\Gamma(w_{i+2}, u)\Gamma(x, w_{i+2})$$

if w_iu is an edge in G; otherwise

$$(x, w_i)\Gamma(w_i, v)\Gamma(v, u)\Gamma(u, w_{i+2})\Gamma(w_{i+2}, x).$$

Case 2. Assume that there exist vertices $v \in N[w_i] - N[x]$ and $u \in N[x] - N[w_{i+2}]$. Then $u, v \in H$. Thus

$$(x, w_i)\Gamma(u, x)\Gamma(x, w_{i+2})$$

if w_iu is not an edge in G; or

$$(x, w_i)\Gamma(w_i, v)\Gamma(v, w_{i+1})\Gamma(w_{i+2}, v)\Gamma(x, w_{i+2})$$

if vw_{i+2} is an edge in G; otherwise

$$(x, w_i)\Gamma(w_i, v)\Gamma(w_{i+2}, w_i)\Gamma(w_i, u)\Gamma(u, w_{i+1})\Gamma(x, u)\Gamma(w_{i+2}, x).$$

Case 3. Assume that there exists a vertex $v \in N[x] - \{N[w_i] \cup N[w_{i+2}]\}$. Then

$$(x, w_i)\Gamma(v, x)\Gamma(x, w_{i+2}).$$

Assume that there exist vertices $v \in N[x] - N[w_i]$ and $u \in N[x] - N[w_{i+2}]$ where $u \neq v$. Then we have

$$(x, w_i)\Gamma(v, x)\Gamma(v, w_{i+1})\Gamma(w_{i+2}, v)\Gamma(w_i, w_{i+2})\Gamma(u, w_i)\Gamma(w_{i+1}, u)\Gamma(x, u)\Gamma(w_{i+2}, x).$$

Case 4. Assume that there exist vertices in $N[x] - N[w_i]$ and in $N[w_{i+2}] - N[x]$. This is similar to Case 2.

To complete the proof that all edges of G in G_i are in the same implication class, consider an edge xy of G where $x \in S$ and $y \in H$. Since G is reduced, we have $N[x] \neq N[y]$. If there is a vertex $z \in S$ such that $z \in N[x] - N[y]$, then xy and xz are in the same implication class and xz lies in S. If there is a vertex $z \in N[y] - N[x]$ then xy and yz are in the same implication class and yz lies in H.

Finally we show that all edges of G between any two fixed G_i and G_j, where $i \neq j$, are in the same implication class. Let xz and yw be two edges of G between G_i and G_j, where $x, y \in G_i$ and $z, w \in G_j$. Since z and w are in the same connected component of $\overline{G}$, there is a path in $\overline{G}_j$ from w to w. Let $z = v_1 \sim v_2 \sim \ldots \sim v_t = w$ be such a path. Then xv_i is an edge of G for each $i = 1, 2, \ldots, t$. We also notice that xv_i and xv_{i+1} are in the same implication class for each $i = 1, 2, \ldots, t-1$. Hence xz and xw are in the same implication class. A similar argument applies to show that xz and yz are in the same implication class. Hence xz and yw are in the same implication class. □

Proposition 2.3.8 completely describes all implication classes of a reduced local-tournament-orientable graph G for which $\overline{G}$ is bipartite.

Suppose that G is a reduced local-tournament-orientable graph which contains a vertex v of degree $n-1$. Then by Corollary 2.1.11, $\overline{G}$ is a bipartite graph. So from now on we need to consider a reduced graph G for which $\Delta(G) \leq n-2$ and $\overline{G}$ is not bipartite. In addition (in view of Proposition 2.3.3) we may assume that G can *only* be oriented as a *strong* local tournament.

By Theorem 2.2.9, we know that G admits a round enumeration, that is, the vertices of G can be circularly ordered $v_1, v_2, \ldots, v_n$ so that

$$N(v_i) = \{v_{i-1}, v_{i-2}, \ldots, v_{i-l}\} \cup \{v_{i+1}, v_{i+2}, \ldots, v_{i+r}\},$$

where $\{v_{i-1}, v_{i-2}, \ldots, v_{i-l}\}$ and $\{v_{i+1}, v_{i+2}, \ldots, v_{i+r}\}$ induce complete subgraphs of G for each vertex v_i. We shall call v_i and v_{i+1} *consecutive* vertices (the subscript addition is modulo n). Note that all consecutive vertices are adjacent since G can not be oriented as a non-strong local tournament.

If v_iv_j is the left-most wave at v_i, then $\{v_j, v_{j+1}, \ldots, v_i\}$ induces a complete subgraph. Similarly if v_iv_k is the right-most wave at v_i, then $\{v_i, v_{i+1}, \ldots, v_k\}$ induces a complete subgraph. Since $\Delta(G) \leq n-2$, there exists a non-neighbour of v_i between v_k and v_j.

Lemma 2.3.9 *Suppose that G is a reduced connected graph for which $\overline{G}$ is not bipartite. Suppose that G is orientable as a strong local tournament and is not orientable as a non-strong local tournament. Let $v_1, v_2, \ldots, v_n$ be a round enumeration of G. If the left-most wave and the right-most wave at each fixed vertex of G are in* *the same implication class, then all left-most waves and all right-most waves at all vertices of G are in one implication class.*

Proof: It suffices to show that the left-most waves and the right-most waves at two consecutive vertices v_i and v_{i+1} are in the same implication class. We shall only prove the case when $i = 1$. A similar proof applies for $i \neq 1$.

Let v_1v_i and v_1v_j be the left-most wave and the right-most wave, respectively, at v_1, and let v_2v_k and v_2v_l be the left-most wave and the right-most wave, respectively, at v_2. Since $N[v_1] \neq N[v_2]$, we have that $j \neq l$ if $i = k$. Similarly we have $i \neq k$ if $j = l$. We consider the following cases.

Case 1. If $i = k$ and $j \neq l$, then $(v_1, v_i)\Gamma(v_i, v_{i-1})\Gamma(v_2, v_k)$. Hence the left-most wave at v_1 is in the same implication class as the left-most wave at v_2.

Case 2. If $j = l$ and $i \neq k$, then $(v_1, v_j)\Gamma(v_j, v_{j+1})\Gamma(v_2, v_l)$. Hence the right-most wave at v_1 is in the same implication class as the right-most wave at v_2.

Case 3. If $i \neq k$ and $j \neq l$, then $j < i$, $l \leq n$. If $i \leq l$, then $deg(v_i) = n - 1$ because $\{v_i, \ldots, v_n, v_1\}$ and $\{v_2, \ldots, v_i\}$ induce complete subgraphs of G, contradicting our hypothesis. Thus $j < l < i$. Hence $(v_i, v_1)\Gamma(v_1, v_2)\Gamma(v_2, v_l)$ and so the left-most wave at v_1 is in the same implication class as the right-most wave of v_2.

Therefore the left-most waves and the right-most waves at v_1 and v_2 are in the same implication class. □

Proposition 2.3.10 *Suppose that G is a reduced connected graph for which $\overline{G}$ is not bipartite. Suppose that G is orientable as a strong local tournament and is not orientable as a non-strong local tournament. Then the edge set of G forms one implication class.*

Proof: Let $v_1, v_2, \ldots, v_n$ be a round enumeration of G. Consider the vertex v_1. Suppose that v_1v_i and v_1v_j, where $i < j$, are two arbitrary edges incident with v_1 (not necessarily waves) such that $< \{v_1, v_2, \ldots, v_i\} >$, $< \{v_j, v_{j+1}, \ldots, v_n, v_1\} >$, and $< \{v_i, v_{i+1}, \ldots, v_j\} >$ are complete. We claim that v_1v_i and v_1v_j are in the same implication class.

Since $\Delta(G) \leq n - 2$, there exists a vertex which is not adjacent to v_j. Choose such a vertex with the greatest subscript, say v_k. Then $1 < k < i$ and v_j is adjacent to v_{k+1}. Thus $(v_j, v_1)\Gamma(v_1, v_k)$ and so v_1v_j and v_1v_k are in the same implication class. We claim that v_k and v_{j+1} are not adjacent. If v_k and v_{j+1} are adjacent, then at least one of the subgraphs $< \{v_k, v_{k+1}, \ldots, v_{j+1}\} >$ and $< \{v_{j+1}, v_{j+2}, \ldots, v_1, \ldots, v_k\} >$ must be complete. However, $< \{v_k, v_{k+1}, \ldots, v_{j+1}\} >$ is not complete since v_k is not adjacent to v_j. Therefore $< \{v_{j+1}, v_{j+2}, \ldots, v_1, \ldots, v_k\} >$ must be complete. A similar argument shows that $< \{v_{k+1}, v_{k+2}, \ldots, v_j\} >$ is complete. This contradicts our assumption that $\overline{G}$ is not bipartite.

Let v_l be the vertex of greatest subscript such that v_l is not adjacent to v_k. Then $j + 1 \leq l \leq n$ and v_k is adjacent to v_{l+1}. So $(v_1, v_k)\Gamma(v_l, v_1)$ and so v_1v_k and v_lv_1 are in the same implication class. Hence v_1v_j and v_1v_l are in the same implication class. Again by the condition that $\overline{G}$ is not bipartite, v_l is not adjacent to v_{k+1}; or else $< \{v_{l+1}, v_{l+2}, \ldots, v_n, v_1, \ldots, v_k\} >$ and $< \{v_{k+1}, v_{k+2}, \ldots, v_l\} >$ are two complete subgraphs covering G and $\overline{G}$ would be bipartite.

If v_l is not adjacent to v_i, then v_1v_l and v_1v_i are in the same implication class and we are done as this implies that v_1v_i and v_1v_j are in the same implication class. If v_l and v_i are adjacent, then again choose a vertex v_m of greatest subscript so that v_m and v_l are not adjacent. Then $k+1 \leq m < i$ and v_1v_l and v_1v_m are in the same implication class. So v_1v_j and v_1v_m are in the same implication class. Notice that $k < m < i$ and so v_m is relatively closer to v_i than v_k. Continuing the above procedure, we will eventually find that v_1v_i and v_1v_j are in the same implication class.

Now we show that the left-most wave and the right-most wave at each fixed vertex are in the same implication class. Without loss of generality, we only consider the left-most wave v_1v_j and the right-most wave v_1v_i at v_1 (Note that our discussion remains the same for every other vertex v_i.) If v_i is not adjacent to v_j, then $(v_1, v_j)\Gamma(v_j, v_1)$ and v_1v_j and v_1v_j are in the same implication class and we are done. Otherwise v_i and v_j are adjacent. Then either $< \{v_i, v_{i+1}, \ldots, v_j\} >$ or $< \{v_j, v_{j+1}, \ldots, v_1, \ldots, v_i\} >$ is complete. If $< \{v_i, v_{i+1}, \ldots, v_j\} >$ is complete, then we have proved that v_1v_i and v_1v_j are in the same implication class. If $< \{v_j, v_{j+1}, \ldots, v_n, v_1, \ldots, v_i\} >$ is complete, let v_iv_l be the right-most wave at v_i and let v_jv_k be the left-most wave at v_j. Then

$i+1 \leq l, k \leq j-1$. If $l \geq k$, then $(v_1, v_i)\Gamma(v_i, v_l)$ and $(v_l, v_j)\Gamma(v_j, v_1)$. We note that v_iv_l and v_lv_j are two edges incident with v_l, and $< \{v_j, \ldots, v_n, v_1, \ldots, v_i\} >$ is complete. We conclude that v_iv_l and v_lv_j are in the same implication class, by using the same arguments made (at the beginning of the proof) for two edges incident with v_1. Hence v_1v_i and v_1v_j are in the same implication class. Suppose that $l < k$. We claim that for any a such that $l < a < j$ the vertex v_a is not adjacent to v_i. In fact, if v_i is adjacent to v_a, then either $< \{v_a, v_{a+1}, \ldots, v_n, \ldots, v_i\} >$ is complete or $< \{v_i, v_{i+1}, \ldots, v_a\} >$ is complete. However we know that v_1 and v_a are two non-adjacent vertices in $< v_a, v_{a+1}, \ldots, v_n, \ldots, v_i >$, and v_i and v_{l+1} are two non-adjacent vertices in $< v_i, v_{i+1}, \ldots, v_a >$, a contradiction. A similar argument applies to show that for each b such that $i < b < k$ the vertex v_b is not adjacent to v_j. Note that $v_l \sim v_{l+1} \sim \ldots \sim v_k$ is a path. So $< \{v_l, v_{l+1}, \ldots, v_k\} >$ is connected. Let

$$v_l = v_{m_1} \sim v_{m_2} \sim \ldots v_{m_h} = v_k$$

be a shortest (v_l, v_k)-path, denoted by $P[v_l, v_k]$, in $< \{v_l, v_{l+1}, \ldots, v_k\} >$. Then we must have $m_1 < m_2 < \ldots < m_h$. The path $P[v_l, v_k]$ is chordless since it is shortest. Hence

$$(v_l, v_{m_1})\Gamma(v_{m_1}, v_{m_2})\Gamma(v_{m_2}, v_{m_3})\Gamma \ldots \Gamma(v_{m_{h-1}}, v_{m_h}).$$

Now $v_{m_1} = v_l$ is not adjacent to v_1. We have

$$(v_1, v_i)\Gamma(v_i, v_{m_1})\Gamma(v_{m_1}, v_{m_2}).$$

Similarly $v_{m_h} = v_k$ is not adjacent to v_1. We have

$$(v_{m_{h-1}}, v_{m_h})\Gamma(v_{m_h}, v_j)\Gamma(v_j, v_1).$$

Therefore $(v_1, v_i)\Gamma(v_j, v_1)$ and v_1v_i is in the same implication class as v_1v_j. By Lemma 2.3.9, all left-most waves and all right-most waves at all vertices are in the same implication class.

Finally we show that any edge belongs to the same implication class as the left-most wave or in the same implication class as the right-most wave at some vertex. Again without loss of generality, we consider an edge v_1v_i incident with v_1. Then either

$$< \{v_1, v_2, \ldots, v_i\} >,$$

or

$$< \{v_i, v_{i+1}, \ldots, v_n, \ldots, v_1\} >$$

is complete. Suppose that $< \{v_1, v_2, \ldots, v_i\} >$ is complete (A similar argument applies if $< \{v_i, v_{i+1}, \ldots, v_n, v_1\} >$ is complete.). Let v_1v_j be the left-most wave at v_1. Then $j > i$. If v_i and v_j are not adjacent, then $(v_1, v_i)\Gamma(v_j, v_1)$ and v_1v_i is in the same implication class as the left-most wave at v_1. Suppose that v_i and v_j are adjacent. Then either

$$< \{v_j, v_{j+1}, \ldots, v_1, \ldots, v_i\} >$$

or

$$< \{v_i, v_{i+1}, \ldots, v_j\} >$$

is complete. Suppose that $< \{v_i, v_{i+1}, \ldots, v_j\} >$ is complete. Then v_1v_i and v_1v_j must be in the same implication class by earlier arguments. Finally suppose that $< \{v_j, v_{j+1}, \ldots, v_1, \ldots, v_i\} >$ is complete. Let v_iv_k be the right-most wave at v_i. Then $i < k < j$. The vertex v_k is not adjacent to v_1 as otherwise $N[v_1] = N[v_i]$, contradicting the fact that G is reduced. Hence $(v_1, v_i)\Gamma(v_i, v_k)$ and v_1v_i is in the same implication class as the right-most wave at v_i. Therefore the edge set of G forms one implication class. □

All implication classes of a reduced local-tournament-orientable graph are therefore completely characterized.

Theorem 2.3.11 *Let G be a reduced connected local-tournament-orientable graph. Suppose that $C_1, C_2, \ldots, C_k$ are the connected components of $\overline{G}$. The one of the following two statements is true.*

- *If $\overline{G}$ is bipartite, then the set of all edges of G within a fixed C_i forms an implication class and the set of all edges of G between two fixed C_i and C_j ($i \neq j$) forms an implication class.*
- *If $\overline{G}$ is not bipartite, then $k = 1$ and the edge set of G forms one implication class.*

Proof: If $\overline{G}$ is bipartite, then the first statement is true according to Proposition 2.3.8. If $\Delta(G) = n-1$, then $\overline{G}$ is bipartite by Corollary 2.1.11 and hence statement 1 is true.

Suppose that $\overline{G}$ is not bipartite. Then $\Delta(G) \leq n-2$. According to Proposition 2.3.3 and 2.3.10, the edge set of G forms one implication class. Assume that $k > 1$. We note that the edges of G between C_1 and C_2 contain an implication class. Since all edges of G are in the same implication class, all edges of G are between C_1 and C_2. Hence $k = 2$, and there is no edge of G within C_1 or C_2. Now we have $|C_1| \leq 2$ and $|C_2| \leq 2$ as otherwise any three vertices in C_1 (or C_2) together with a vertex in C_2 (or C_1) induce a copy of the claw (see Fig. 2.1) in G, contradicting the fact that G is local-tournament-orientable. Therefore $\overline{G}$ is bipartite, contrary to our hypothesis. □

Corollary 2.3.12 *Let D be a connected local tournament with reduced $G(D)$. Then any local-tournament orientation of $G(D)$ is obtained from D by performing partial reversals, possibly followed by a full reversal.*

Proof: Suppose that D' is a local-tournament orientation of $G(D)$. Since $G(D) = G(D')$, an implication class of $G(D)$ is also an implication classes of $G(D')$. Suppose that $C = \{a_1b_1, a_2b_2, \ldots, a_lb_l\}$ is an implication class and suppose that a_i dominates b_i in D for each i. Then, in D', either a_i dominates b_i, or b_i dominates a_i, for each i. If $C_1, C_2, \ldots, C_k$ are the connected components of $\overline{G}$, then by Corollary 2.3.11 one of the following cases occurs:

- C consists of all edges of G within C_i for some i,
- C consists of all edges of G between C_i and C_j for some i and j,
- C consists of all edges of G.

Therefore, by the definitions of a partial reversal and a full reversal, D' is obtained from D by performing partial reversals, possibly followed by a full reversal. □

Corollary 2.3.13 *Suppose that $G = (V, E)$ is a reduced proper circular arc graph with $|V| > 4$ which contains no isolated vertices. Then G is uniquely local-tournament-orientable if and only if both G and $\overline{G}$ are connected.*

Proof: We remark first that a connected graph is a proper circular arc graph if and only if it is local-tournament-orientable. The sufficiency follows from Theorem 2.3.11. To prove the necessity, suppose that G is uniquely local-tournament-orientable. First G must be connected as otherwise each connected component has at least two orientations (one is obtained by the full reversal of the other) and so the total number of local-tournament orientations of G is at least 4. Suppose that $\overline{G}$ is not connected. Let $C_1, C_2, \ldots, C_k$ be connected components of $\overline{G}$ with $k \geq 1$. Then by Theorem 2.3.11, $\overline{G}$ must be bipartite. According to our hypothesis G is uniquely local-tournament-orientable, that is, the edge set of G must form one implication class. We note that the edges of G between C_1 and C_2 contain an implication class. Then all edges of G are between C_1 and C_2. Hence $k = 2$, and there is no edge of G within C_1 or C_2. Now we have $|C_1| \leq 2$ and $|C_2| \leq 2$ as otherwise any three vertices in C_1 (or C_2) together with a vertex in C_2 (or C_1) induce a copy of the claw in G, contradicting the fact that G is local-tournament-orientable. Therefore $|V| \leq 4$, contradicting our hypothesis. □

Let P_3 and C_4 denote a path of length 3 and a cycle of length 4, respectively. Then $\overline{P_3}$ and $\overline{C_4}$ are not connected but both of P_3 and C_4 are uniquely orientable as a local tournament. So the condition $|V| > 4$ in Corollary 2.3.13 is necessary.

We will next analyze the implication classes of a local-tournament-orientable graph G which is not necessarily reduced. First we have the following lemma.

Lemma 2.3.14 *Let G be a connected local-tournament-orientable graph. Suppose that xy and xz are two unbalanced edges and yz is a balanced edge of G. If $\deg(y) \leq n-2$, then $(x,y)\Gamma^*(x,z)$ and hence xy and xz are in the same implication class.*

Proof: It suffices to prove that in any local-tournament orientation of G, x dominates y if and only if x dominates z. Assume to the contrary that $x \rightarrow y$ and $z \rightarrow x$ in some local-tournament orientation D of G. Since xy is an unbalanced edge, either

$N[x] - N[y] \neq \emptyset$ or $N[y] - N[x] \neq \emptyset$. If there is a vertex $u \in N[x] - N[y]$, then u is adjacent to x but not to y and hence not to z (since $N[y] = N[z]$). If $x \rightarrow u$, then x has two non-adjacent out-neighbours u and y. If $u \rightarrow x$, then x has two non-adjacent in-neighbours u and z. Hence $N[x] - N[y] = \emptyset$ and so there is a vertex $w \in N[y] - N[x]$. Then w is adjacent to y and z but not to x. Hence $y \rightarrow w$ and $w \rightarrow z$ in D. An argument similar to the one above, with w replacing x, shows that $N[w] - N[y] = \emptyset$.

Since $deg(y) \leq n - 2$, there exists a vertex v which is not adjacent to y. However there is a path in G from v to y as G is connected. Let $v = v_1 \sim v_2 \sim \ldots \sim v_t = y$ be a shortest path from v to y. Then $t \geq 3$ as v is not adjacent to y. If $v_{t-1} = x$ or $v_{t-1} = w$, then $v_{t-2} \in N[x] - N[y]$ or $v_{t-2} \in N[w] - N[y]$, contradicting the fact $N[x] - N[y] = \emptyset$ and $N[w] - N[y] = \emptyset$. So $v_{t-1} \neq x$ and $v_{t-1} \neq w$. We note that v_{t-1} is adjacent to at least one of x and w as otherwise $\{x, y, w, v_{t-1}\}$ induces a copy of the claw in G and G is not local-tournament-orientable, a contradiction. Without loss of generality, suppose that v_{t-1} is adjacent to w. If v_{t-1} is not adjacent to x, then $y \rightarrow v_{t-1}$ and $v_{t-1} \rightarrow z$ because $z \rightarrow x$ and $N[y] = N[z]$. If $v_{t-2} \rightarrow v_{t-1}$, then v_{t-1} has two non-adjacent in-neighbours v_{t-2} and y, contradicting the fact that D is local-tournament-orientable. If $v_{t-1} \rightarrow v_{t-2}$, then v_{t-1} has two non-adjacent out-neighbours v_{t-2} and z, a contradiction. Hence v_{t-1} is also adjacent to x.

Note that v_{t-2} is adjacent to at least one of x and w as otherwise $\{v_{t-1}, v_{t-2}, x, w\}$ induces a copy of the claw. However if v_{t-2} is adjacent to x or w, then $v_{t-2} \in N[x] - N[y]$ or $v_{t-2} \in N[w] - N[y]$, which contradicts the fact that $N[x] - N[y] = \emptyset$ and $N[w] - N[y] = \emptyset$. □

Theorem 2.3.15 *Let G be a connected local-tournament-orientable graph (not necessarily reduced). Suppose that $C_1, C_2, \ldots, C_k$ are the connected components of $\overline{G}$. Then one of the following two statements is true.*

- *If $\overline{G}$ is bipartite, then all unbalanced edges of G within a fixed C_i form an implication class and all unbalanced edges of G between two fixed C_i and C_j $(i \neq j)$ form an implication class.*

- *If $\overline{G}$ is not bipartite, then $k = 1$ and all unbalanced edges of G form one implication class.*

Proof: Let xy and uv be two unbalanced edges of G, where $x, u \in C_a$ and $y, v \in C_b$ for some a and b with $1 \le a, b \le k$. Then $N[x] \ne N[y]$ and $N[u] \ne N[y]$. Suppose that $\overline{G}$ is bipartite. Assume first that $a = b$. Note that each vertex of degree $n-1$ in G forms a connected component of $\overline{G}$. In other words, if some C_i has at least two vertices, then each vertex of C_i has at most $n-2$ neighbours in G. Suppose that $N[x] = N[u]$ and $N[y] = N[v]$. Then x is adjacent to v, and y is adjacent to u. By Lemma 2.3.14, xy is in the same implication class as xv, and xv is in the same implication class as uv. Thus xy and uv are in the same implication class. Assume $N[x] = N[u]$ and $N[y] \ne N[v]$. (A symmetric argument applies when $N[x] \ne N[u]$ and $N[y] = N[v]$.) Applying Theorem 2.3.11 and Lemma 2.3.14 to a subgraph of G which contains exactly one vertex from each block of G and contains the vertices x, y, and v, we conclude that xy is in the same implication class as xv. From above, we have that xv is in the same implication class as uv. Hence xy and uv are in the same implication class. Assume that the closed neighbourhoods $N[x], N[y], N[u]$, and $N[v]$ are mutually distinct. Then applying Theorem 2.3.11 to a subgraph of G which contains exactly one vertex from each block of G and contains the vertices x, y, u, and v, we conclude that xy is in the same implication class as uv.

Assume now that $a \ne b$. Suppose that each of C_a and C_b consists of a single vertex of degree $n-1$. Then $x = u$ and $y = v$. Thus the conclusion follows trivially. Suppose that one of C_a and C_b, say C_a, consists of a single vertex of degree $n-1$. Then $x = u$. If $N[y] = N[v]$, then by Lemma 2.3.14, xy is in the same implication class as uv. Suppose that $N[y] \ne N[v]$. Applying Theorem 2.3.11 to a subgraph of G which contains a vertex from each block of G and contains the vertices x, u, and v, we conclude that xy is in the same implication class as uv. For the case when none of C_a and C_b consists of a single vertex of degree $n-1$, the discussions are similar to the case when $a = b$.

Finally suppose that $\overline{G}$ is not bipartite. Applying Theorem 2.3.11 to a subgraph of G which contains a vertex from each block of G, we conclude that $k = 1$. Note that none of x, y, u, v has degree $n-1$. Hence the discussions are similar again as for the case when $a = b$. □

Proof of Theorem 2.3.2: Let D be a connected oriented graph. Suppose that D is obtained from some round oriented graph R by substituting a tournament T_x for each vertex x of R. Let $x_1, x_2, \ldots, x_l$ be a round enumeration of R. To see that D is a local tournament, let x be a vertex of D and let y and z be two out-neighbours of x. Then $x \in T_{x_i}$, $y \in T_{x_j}$, and $z \in T_{x_k}$ for some $1 \leq i, j, k \leq l$. We show that y and z are adjacent. If $i = j$, then x and y have the same closed neighbourhood and hence y is adjacent to z (as x is adjacent to z). A similar argument applies if $i = k$. If $j = k$, then y and z are adjacent because they are in the same tournament T_j. Assume that i, j, k are mutually distinct. Without loss of generality, assume that x_i, x_j, x_k are three vertices of R listed in clockwise circular order of the round enumeration. Since $x \rightarrow \{y, z\}$, we have that $x_i \rightarrow \{x_j, x_k\}$. By the definition of a round enumeration, we know that $x_j \rightarrow x_k$. Thus $y \rightarrow z$ in D. Hence the outset of x induces a tournament in D. A similar argument applies to show that any two in-neighbours of x are adjacent. Therefore D is a local tournament.

Suppose in turn that D is a local tournament. Let $T_1 \cup T_2 \cup \ldots \cup T_p$ be a partition of D into blocks. Then each T_i is a tournament and if $i \neq j$ then T_i is either completely adjacent to T_j or completely non-adjacent to T_j.

Let $\{x_1, x_2, \ldots, x_l\}$ be a set of vertices such that $x_i \in T_i$ for each $i = 1, 2, \ldots, l$. Suppose that R is the subgraph of D induced by $\{x_1, x_2, \ldots, x_l\}$. Since D is a connected local tournament, we know that R is also a connected local tournament. In fact R is reduced because distinct vertices of R have distinct closed neighbourhoods. By Theorem 2.2.9, R admits a round enumeration. Without loss of generality, assume that $x_1, x_2, \ldots, x_l$ is a round enumeration of R.

Suppose that $\Delta(D) \leq n - 2$. It is implied by Lemma 2.3.14 that $T_i \rightarrow T_j$ if and only if $x_i \rightarrow x_j$. Thus D is obtained from R by substituting a tournament T_i for x_i for each $i = 1, 2, \ldots, l$. To prove the second assertion of Theorem 2.3.2, let D' be any local-tournament orientation of $G(D)$. Then D' can be partitioned into vertex disjoint subgraphs $T_1', T_2', \ldots, T_l'$ where $G(T_i') = G(T_i)$ for each $i = 1, 2, \ldots, l$. We also see that each T_i' is a tournament and if $i \neq j$ then T_i' is either completely adjacent to T_j' or completely non-adjacent to T_j'. Moreover if T_i' and T_j' are completely adjacent then either $T_i' \rightarrow T_j'$ or $T_j' \rightarrow T_i'$ by Lemma 2.3.14. First we see that T_i' is obtained from T_i by reversing some arcs in T_i. Suppose that $T_i' = T_i$. Let $\{y_1, y_2, \ldots, y_l\}$ be a set of vertices where $y_i \in T_i'$ and let R' be the subgraph of D' induced by $\{y_1, y_2, \ldots, y_l\}$. Then $y_i \rightarrow y_j$ if and only if $T_i' \rightarrow T_j'$. It is easy to see that the mapping $f : x_i \rightarrow y_i$ is an isomorphism between $G(R)$ and $G(R')$. Then by Corollary 2.3.12, R' is obtained from R by performing partial reversals, possibly followed by a full reversal. Hence D' is obtained from D by performing partial reversals, possibly followed by a full reversal.

Suppose in turn that $\Delta(D) = n - 1$. Without loss of generality, assume that T_1 is induced by the set of vertices of degree $n - 1$. Thus if $i \neq 1$ then every vertex of T_i has at most $n - 2$ neighbours. By Lemma 2.3.14 we have $(x, y)\Gamma(x, z)$ for any two vertices y and z in T_i. Hence by Lemma 2.1.8 either $x \rightarrow \{y, z\}$ or $\{y, z\} \rightarrow x$. Therefore either $x \rightarrow T_i$ or $T_i \rightarrow x$. Let $z_1 \in T_1$ and let R'' be the subgraph of D induced by $\{z_1, x_2, \ldots, x_p\}$. It is easy to verify that the mapping $f : z_1 \rightarrow x_1;\ x_i \rightarrow x_i (i \geq 2)$ is an isomorphism between $G(R)$ and $G(R'')$. Hence, by Corollary 2.3.12, R'' is isomorphic, under f, to R or to an oriented graph obtained from R by performing partial reversals of R. Note that R'' differs from R in only one vertex, i.e., $R'' - z_1 = R - x_1$. So each possible partial reversal of R reverses some arcs incident with x_1. Hence D is obtained from R by first substituting T_i for each x_i and then by performing partial reversals (each partial reversal is performed by reversing some arcs incident with one vertex of T_1). The second assertion of Theorem 2.3.2 can be proved analogously as for the case $\Delta(D) \leq n - 2$. □

Chapter 3

The Lexicographic Method

3.1 Local-bicomplete Orientability

A transitively orientable graph is also called *a comparability graph* (cf. [25, 32, 33, 40, 71]). Since every transitive oriented graph is locally bicomplete, all transitively orientable graphs are local-bicomplete-orientable. It was first observed by Ghouilà-Houri, in different terminology, that the converse of the statement is also true (cf. [31]).

Theorem 3.1.1 *A graph is local-bicomplete-orientable if and only if it is transitively orientable.* □

In Chapter 2, we proved that a graph G is a proper circular arc graph if and only if the associated characteristic graph G^* is 2-colourable. From this result, a simple algorithm was obtained there to recognize proper circular arc graphs. In a similar way, we will define in this chapter another associated graph G^+ of G and prove that G is a comparability graph if and only if G^+ is 2-colourable. This will also yield a simple algorithm to recognize comparability graphs. (We have recently learned this result was also formulated and proved by Ghouilà-Houri [31]. However our proof is simpler and additionally yields the algorithm below.)

Let $G = (V, E)$ be a graph and recall from Chapter 2 the notation $F(G)$ for the set of all ordered pairs (u, v) such that uv is an edge of G. Note that in an orientation of G, each edge $uv \in E$ gives rise to two ordered pairs (u, v) and (v, u) of $F(G)$. In other words, by choosing one of (u, v) or (v, u) for each edge uv of G we get an orientation of G.

We now define the *characteristic graph* G^+ with the vertex set $F(G)$ and the adjacency defined by the following: Each $(u, v) \in F(G)$ is adjacent to (v, u), to any $(w, u) \in F(G)$ such that $v \neq w$ and $vw \notin E$, to any $(v, w) \in F(G)$ such that $u \neq w$ and $uw \notin E$, and to no other vertex of G^+.

Theorem 3.1.2 *A graph $G = (V, E)$ admits a local-bicomplete orientation if and only if the characteristic graph G^+ is 2-colourable.*

Moreover, if G^+ is 2-coloured with A being a colour class, then $D = (V, A)$ is a local-bicomplete orientation of G.

Proof: Suppose that D is a local-bicomplete orientation of G. We colour the vertices of G^+ with two colours μ and τ in the following way: Colour a vertex (u, v) by μ if u dominates v, and by τ if v dominates u in D. Let (x, y) and (x', y') be two adjacent vertices of G^+. It is easy to see that x dominates y if and only if y' dominates x' in D. Hence (x, y) and (x', y') are coloured with different colours. Therefore G^+ is 2-colourable.

Suppose now that G^+ is 2-coloured with A being a colour class. We prove that $D = (V, A)$ is a local-bicomplete orientation of G. Since, for each $(u, v) \in F(G)$, (u, v) and (v, u) are adjacent in G^+, exactly one of (u, v) and (v, u) belongs to A. Thus D is an orientation of G. To show that D is locally bicomplete, let u, v, and w be three vertices of G such that v and w are two non-adjacent neighbours of u. Then $(u, v), (w, u) \in F(G)$ are adjacent in G^+ (and (v, u) and (u, w) are adjacent in G^+). Hence at most one of (u, v) and (w, u) (and at most one of (v, u) and (u, w)) belongs to A. Therefore D is a local-bicomplete orientation of G. □

Theorem 3.1.2 proves the correctness of the following algorithm for finding local-bicomplete orientations.

Algorithm 3.1.3 *Let $G = (V, E)$ be a graph.*

Step 1. Construct the characteristic graph G^+ of G.

Step 2. If G^+ is not 2-colourable, then G is not local-bicomplete-orientable.

Step 3. If G^+ is 2-colourable, then find any 2-colouring of G^+ and obtain a local-bicomplete orientation $D = (V, A)$ of G where A is a colour class of G^+. □

Theorem 3.1.4 *There is an $O(m\Delta)$ time algorithm to recognize local-bicomplete-orientable graphs and to find such an orientation if one exists.*

Proof: The graph G^+ has $O(m)$ vertices, $O(\Sigma_{uv \in E} deg(u) + deg(v)) = O(m\Delta)$ edges and it can be constructed in time $O(m\Delta)$. In the same time we can test, by breath-first search, whether it is 2-colourable, and find a 2-colouring of G^+. □

Corollary 3.1.5 *There is an $O(m\Delta)$ algorithm to recognize comparability graphs.*

Proof: This is immediate from Theorem 3.1.1 and 3.1.4. □

Let $(u, v), (x, y) \in F(G)$. We say that (u, v) *pushes* (x, y), denoted by $(u, v)\Psi(x, y)$, if one of the following conditions is satisfied.

- $u = x$ and $v = y$;
- $u = x$, $v \neq y$, and $vy \notin E$;
- $v = y$, $u \neq x$, and $ux \notin E$.

It is obvious that $(u, v)\Psi(x, y)$ if and only if (u, v) is adjacent to (y, x) (or (v, u) is adjacent to (x, y)) in G^+. We say that (u, v) *controls* (x, y), denoted by $(u, v)\Psi^*(x, y)$, if there exist $(u_1, v_1), (u_2, v_2), \ldots, (u_k, v_k) \in F(G)$ so that

$$(u, v) = (u_1, v_1)\Psi(u_2, v_2)\Psi \ldots \Psi(u_k, v_k) = (x, y).$$

Proposition 3.1.6 *For any graph G, the binary relation Ψ^* on $F(G)$ is an equivalence relation.* □

According to Proposition 3.1.6, the binary relation Ψ^* partitions $F(G)$ into equivalence classes. We call each of these equivalence classes a Ψ^*-class.

Lemma 3.1.7 *Let $D = (V, A)$ be a local-bicomplete orientation of $G = (V, E)$. If $(u, v)\Psi^*(x, y)$ for some (u, v), $(x, y) \in F(G)$, then $u \to v$ if and only if $x \to y$ in D.*

Proof: We prove that if $(u, v)\Psi(x, y)$ for some (u, v), $(x, y) \in F(G)$, then $u \to v$ if and only if $x \to y$. The general proof can be done by induction.

If $u = x$ and $v = y$, then the conclusion follows trivially. Suppose that $v = y$, $u \neq x$, and $ux \notin E$. If $u \to v$ and $y \to x$ in D, then $u \in I(v)$, $x \in O(v)$, and u is not adjacent to x, contradicting the fact that D is locally bicomplete. If $v \to u$ and $x \to y$ in D, then $x \in I(v)$, $u \in O(v)$, and x is not adjacent to u, contradicting the fact that D is locally bicomplete. A similar proof applies when $u = x$, $v \neq y$, and $vy \notin E$. □

Theorem 3.1.8 *A graph G admits a local-bicomplete orientation if and only if there is no $(u, v) \in F(G)$ such that $(u, v)\Psi^*(v, u)$.*

Proof: The necessity follows immediately from Lemma 3.1.7. For the sufficiency, suppose that there is no $(u, v) \in F(G)$ with $(u, v)\Psi^*(v, u)$. We apply the following procedure to obtain an orientation D of G. Arbitrarily pick an edge uv E which has not been oriented and let $x \to y$ in D for all (x, y) such that $(u, v)\Psi^*(x, y)$. Continue the procedure until there are no unoriented edges left. Since there is no $(u, v) \in F(G)$ with $(u, v)\Psi^*(v, u)$, each edge of G is assigned precisely one orientation. Thus D is an orientation of G. It suffices to show that D is locally bicomplete. Suppose to the contrary that D is not locally bicomplete. Then there exists a vertex x such that there is a vertex in $y \in O(x)$ and a vertex $z \in I(x)$ such that y is not adjacent to z. Hence $(x, y)\Psi(x, z)$. By the above procedure, we have that $x \to y$ if and only if $x \to z$ in D. Since $x \to y$ in D, we have $x \to z$ in D. Therefore in D we have both $x \to z$ and

$z \rightarrow x$, a contradiction. □

The proof of Theorem 3.1.8 gives an alternative implementation of Algorithm 3.1.3, by working directly on the graph G.

Corollary 3.1.9 *A graph G is local-bicomplete-orientable if and only if $B \cap B^{-1} = \emptyset$ for each Ψ^*-class B.*

Proof: This follows from Proposition 3.1.6, Theorem 3.1.8, and the fact that $B \cap B^{-1} \neq \emptyset$ if and only if B contains both (u, v) and (v, u) for some $(u, v) \in F(G)$. □

Corollary 3.1.10 *If G is local-bicomplete-orientable, then G contains no chordless cycle of odd length as an induced subgraph.*

Proof: It suffices to show that a chordless of odd length is not local-bicomplete-orientable. Assume that $x_1 \sim x_2 \sim \ldots x_r \sim x_1$ is a chordless cycle where r is odd. Since $(x_i, x_{i-1})\Psi(x_i, x_{i+1})$ for each $i = 1, 2, \ldots, r$, we have that $(x_1, x_2)\Psi(x_2, x_1)$ because r is odd. Hence the result follows from Theorem 3.1.8. □

Proposition 3.1.11 *Let G be a local-bicomplete-orientable graph and suppose that G^+ is coloured with two colours. Then each Ψ^*-class consists of all vertices of one colour in one component of G^+.*

Proof: Suppose that A is a Ψ^*-class. For any two elements (u, v) and (x, y) from A, by the definition of a Ψ^*-class, there exist $x_1y_1, x_2y_2, \ldots, x_iy_i$ such that

$$(u, v) = (x_1, y_1)\Psi(x_2, y_2)\Psi \ldots \Psi(x_i, y_i) = (x, y).$$

Hence, in G^+,

$$(u, v) = (x_1, y_1) \sim (y_1, x_1) \sim (x_2, y_2) \sim \ldots \sim (x_i, y_i) = (x, y)$$

is a path of even length from (u, v) to (x, y). Therefore (u, v) and (x, y) are in the same component and must be coloured with the same colour.

On the other hand, suppose that (u, v) and (x, y) are coloured with the same colour and are in the same component of G^+. Then there is a path of even length from (u, v) to (x, y). Assume that

$$(u, v) = (a_1, b_1) \sim (a_2, b_2) \sim \ldots \sim (a_j, b_j) = (x, y),$$

such a path. Then

$$(u, v) = (a_1, b_1)\Psi(b_2, a_2)\Psi \ldots \Psi(a_j, b_j) = (x, y).$$

Hence (u, v) and (x, y) are in the same Ψ^*-class. □

Suppose that G is local-bicomplete-orientable and suppose that G^+ is coloured with μ and τ. If B is a set of vertices coloured with μ in one component of G^+, then B^{-1} is the set of vertices coloured with τ in the same component as the vertices of B. Hence both B and B^{-1} are independent in G^+.

Note that by switching the two colours of vertices in a component of G^+ we get a new 2-colouring of G^+. Therefore if we let

$$F(G) = B_1 \cup B_2 \cup \ldots \cup B_t \cup B_1^{-1} \cup B_2^{-1} \cup \ldots \cup B_t^{-1}$$

be the decomposition of $F(G)$ into Ψ^*-classes, then each B_i (and each B_i^{-1}) is an independent set in G^+ and each $B_i \cup B_i^{-1}$ induces a component of G^+ for each $i = 1, 2, \ldots, t$. Moreover, by Algorithm 3.1.3, a local-bicomplete orientation of G can be obtained by choosing the arc set to be $A_1 \cup A_2 \cup \ldots \cup A_t$ where $A_i = B_i$ or B_i^{-1} for each $i = 1, 2, \ldots, t$. In fact, it is not difficult to see that this gives all possible local-bicomplete orientations of G.

Corollary 3.1.12 *A graph G is uniquely local-bicomplete-orientable if and only if G^+ is a connected bipartite graph.* □

We close this section by presenting the following theorem.

Theorem 3.1.13 *The following statements are equivalent for a graph G:*

1. *G is a comparability graph;*

2. *G is local-bicomplete-orientable;*

3. *G is transitively orientable;*

4. *G^+ is 2-colourable;*

5. *$B \cap B^{-1} = \emptyset$ for each Ψ^*-class.*

Proof: The equivalence between statements 1 and 3 is from the definition of a comparability graph. The equivalence between statements 2 and 3 is basically Theorem 3.1.1. The equivalence between statements 2 and 4 is precisely Theorem 3.1.2. Finally the equivalence between statements 2 and 5 is just Corollary 3.1.9. □

3.2 Orientation Algorithms

In 1971, Pnueli, Lempel, and Even [63] gave an $O(m\Delta)$ time algorithm to recognize comparability graphs and to calculate transitive orientations. This algorithm relies on a deep analysis of structures in comparability graphs and it is quite complicated. Here we provide a simple algorithm to solve the same problem. Our algorithm also runs in time $O(m\Delta)$ and makes use of a novel lexicographic method. Using the same method, we will obtain $O(m\Delta)$ algorithms to recognize proper interval graphs and proper circular arc graphs, and to calculate acyclic local-tournament orientations and local-transitive-tournament orientations.

Let G be a graph. Suppose that the vertices of G are enumerated as $v_1, v_2, \ldots, v_n$. In order to describe our algorithm, we define a lexicographic order among all subsets of the vertex set of G. We say that $\{v_i\}$ is lexicographically less than $\{v_j\}$, denoted by $\{v_i\} \ll \{v_j\}$, if $i < j$. In general, let X and Y be two sets of vertices of G of size k. Write

$$X = \{x_1, x_2, \ldots, x_k\},\ \{x_1\} \ll \{x_2\} \ll \ldots \ll \{x_k\}$$

and

$$Y = \{y_1, y_2, \ldots, y_k\},\ \{y_1\} \ll \{y_2\} \ll \ldots \ll \{y_k\}.$$

We say that X is lexicographically less than Y, denoted by $X \ll Y$, if there exists an r such that $1 \le r \le k$, $\{x_t\} = \{y_t\}$ for all $t < r$, and $\{x_r\} \ll \{y_r\}$. Suppose that $\mathcal{L}$ is a collection of sets of size k. Then $X \in \mathcal{L}$ is called *lexicographically smallest* in $\mathcal{L}$ if $X \ll Y$ for any $Y \in \mathcal{L}$ such that $Y \neq X$.

3.2.1 The Transitive Orientation Algorithm for Comparability Graphs

We consider the following algorithm for finding transitive orientations.

Algorithm 3.2.1 *Let $G = (V, E)$ be a graph.*

Step 1. Construct the characteristic graph G^+.

Step 2. If G^+ is not 2-colourable, then G is not a comparability graph.

Step 3. If G^+ is 2-colourable, then find a special 2-colouring of G^+ with colours μ and τ by always first assigning μ to the lexicographically smallest uncoloured vertex (x, y), and completing the unique 2-colouring of the corresponding component of G^+.

Step 4. Let A be the set of vertices of G^+ which are coloured with μ, and obtain a transitive orientation $D = (V, A)$ of G. □

Suppose that G is a comparability graph. By Theorems 3.1.1 and 3.1.2 we know that G^+ is 2-colourable. For each vertex (u, v) of G^+, we use $\sigma[(u, v)]$ to denote the set of vertices which have even distance from (u, v) in G^+. Then for every $(x, y) \in \sigma[(u, v)]$

we must have $\sigma[(x, y)] = \sigma[(u, v)]$. By Proposition 3.1.11, $\sigma[(u, v)]$ is precisely the Ψ^*-class which contains (u, v). So if $\sigma[(x, y)] = \sigma[(u, v)]$, then in any locally bicomplete orientation of G, $x \to y$ if and only if $u \to v$. In other words, in any 2-colouring of G^+ all vertices in $\sigma[(u, v)]$ must be coloured with the same colour. According to this notation, Step 3 of Algorithm 3.2.1 can be interpreted as follows: If G^+ is 2-colourable, then find a special 2-colouring of G^+ with two colours μ and τ by always first assigning colour μ to the lexicographically smallest uncoloured vertex (x, y), as well as to all vertices in $\sigma[(x, y)]$, and colouring all vertices in $\sigma[(y, x)]$ by τ. Note that if (x, y) is the lexicographically smallest pair chosen by Step 3, then $\{x\} \ll \{y\}$.

The following lemma is crucial for proving the correctness of Algorithm 3.2.1.

Lemma 3.2.2 *Suppose that D is locally bicomplete and suppose that $a \to b \to c \to a$ is a directed triangle in D. If $\sigma[(b, c)] = \sigma[(b', c')]$ for some $(b', c') \in F(G)$, then $\sigma[(a, b)] = \sigma[(a, b')]$ and $\sigma[(c, a)] = \sigma[(c', a)]$.*

Proof: Since $\sigma[(b, c)] = \sigma[(b', c')]$, we know that (b, c) controls (b', c'). Then there exist $(b_1, c_1), (b_2, c_2), \ldots, (b_l, c_l) \in F(G)$ for some l such that

$$(b, c) = (b_1, c_1)\Psi(b_2, c_2)\Psi \ldots \Psi(b_l, c_l) = (b', c').$$

Without loss of generality, we assume that $(b_i, c_i) \neq (b_{i+1}, c_{i+1})$ for each $i = 1, 2, \ldots, l-1$. Since b dominates c, we know that b_i dominates c_i for each $i = 1, 2, \ldots, l$ (see Lemma 3.1.7). We prove that $\sigma[(a, b)] = \sigma[(a, b_i)]$ and $\sigma[(c, a)] = \sigma[(c_i, a)]$ for each $i = 1, 2, \ldots, l$.

It is trivial when $i = 1$. Assume that $\sigma[(a, b)] = \sigma[(a, b_i)]$ and $\sigma[(c, a)] = \sigma[(c_i, a)]$ for some $1 \leq i < l$. Then $a \to b_i$ and $c_i \to a$ in D. Since $(b_i, c_i)\Psi(b_{i+1}, c_{i+1})$, by the definition of Ψ, either $c_{i+1} = c_i$, $b_{i+1} \neq b_i$, and b_{i+1} is not adjacent to b_i, or $b_{i+1} = b_i$, $c_{i+1} \neq c_i$, and c_{i+1} is not adjacent to c_i. In the former case, we have that $c_{i+1}a = c_i a$. Hence $\sigma[(c_{i+1}, a)] = \sigma[(c_i, a)]$, and so $\sigma[(c, a)] = \sigma[(c_{i+1}, a)]$. Since $b_{i+1} \to c_{i+1} \to a$ in D, which is locally bicomplete, b_{i+1} is adjacent to a. Hence $\sigma[(a, b_{i+1})] = \sigma[(a, b_i)]$ because b_i is not adjacent to b_{i+1}. Therefore $\sigma[(a, b_{i+1})] = \sigma[(a, b)]$ because $\sigma[(a, b)] = \sigma[(a, b_i)]$. A similar discussion applies in the latter case. □

Theorem 3.2.3 *If G is local-bicomplete-orientable, then Algorithm 3.2.1 correctly finds a transitive orientation $D = (V, A)$ of G.*

Proof: By Theorem 3.1.2, we know that $D = (V, A)$ is a locally bicomplete orientation of G. So it suffices to show that D is transitive. Suppose that D is not transitive. Then there exists a vertex $x \in V(D)$ such that an in-neighbour of x is dominated by an out-neighbour of x, that is, $y \rightarrow z$ for some $y \in O(x)$ and $z \in I(x)$. Hence D contains a directed triangle $x \rightarrow y \rightarrow z \rightarrow x$.

Let $\{a, b, c\}$ be the lexicographically smallest set of size 3 which induces a directed triangle in D. Since $\{a, b, c\}$ induces a directed triangle, there exist two vertices $x, y \in \{a, b, c\}$ such that $x \rightarrow y$ and $\{y\} \ll \{x\}$. Without loss of generality, assume that b and c are two such vertices, that is, $b \rightarrow c$ and $\{c\} \ll \{b\}$. Then there exists $(b', c') \in F(G)$ such that (b', c') was the lexicographically smallest one chosen in Step 3 of Algorithm 3.2.1, such that $\sigma[(b, c)] = \sigma[(b', c')]$. Hence $\{b', c'\} \ll \{b, c\}$. By Lemma 3.2.2 we have $\sigma[(a, b)] = \sigma[(a, b')]$ and $\sigma[(c, a)] = \sigma[(c', a)]$. So $a \rightarrow b'$ and $c' \rightarrow a$ in D. Therefore the set $\{a, b', c'\}$ induces a directed triangle in D and is lexicographically less than $\{a, b, c\}$, contradicting the choice of $\{a, b, c\}$. □

We now provide a simple proof of Ghouilà-Houri's Theorem as follows.

Proof of Theorem 3.1.1: The sufficiency is obvious. To proof the necessity, suppose that G is local-bicomplete-orientable. Then by Theorem 3.2.3, G is transitively orientable. □

3.2.2 The Local-transitive-tournament Orientation Algorithm for Proper Circular Arc Graphs

Now we turn to proper circular arc graphs and their related local-tournament orientations and local-transitive-tournament orientations. Theorem 2.1.1 assures that a proper circular arc graph is local-tournament-orientable. We shall prove that if a graph is local-tournament-orientable then it is also local-transitive-tournament-orientable. We have seen from Theorem 2.2.6 that a proper circular arc representation of G can be obtained in time $O(m+n)$ from a local-transitive-tournament orientation of G. So it is important to understand how to obtain local-transitive-tournament orientations of a proper circular arc graph.

Consider the following algorithm for finding local-transitive-tournament orientations.

Algorithm 3.2.4 *Let $G = (V, E)$ be a connected graph.*

Step 1. Construct the characteristic graph G^*.

Step 2. If G^* is not 2-colourable, then G is not a proper circular arc graph.

Step 3. If G^* is 2-colourable, then find a special 2-colouring of G^* with colours μ and τ by always first assigning μ to the lexicographically smallest uncoloured vertex (x, y), and completing the unique 2-colouring of the corresponding component of G^*.

Step 4. Let A be the set of vertices of G^* which are coloured with μ, and obtain a local-transitive-tournament orientation $D = (V, A)$ of G. □

Suppose that G is a proper circular arc graph. By Theorems 2.1.1 and 2.1.3 we know that G^* is 2-colourable. For each vertex (u, v) of G^*, we use $\rho[(u, v)]$ to denote the set of vertices which have even distance from (u, v) in G^*. Then for every $(x, y) \in \rho[(u, v)]$ we must have $\rho[(x, y)] = \rho[(u, v)]$. By Proposition 2.1.13, $\rho[(u, v)]$ is precisely the Γ^*-class which contains (u, v). So if $\rho[(x, y)] = \rho[(u, v)]$, then in any local-tournament orientation of G, $x \rightarrow y$ if and only if $u \rightarrow v$. In other words, in any 2-colouring of G^* the vertices in $\rho[(u, v)]$ must be coloured with the same colour. According to this notation, Step 3 of Algorithm 3.2.4 can be interpreted as follows: If G^* is 2-colourable, then find a special 2-colouring of G^* with two colours μ and τ by always first assigning colour μ to the lexicographically smallest uncoloured vertex

(x, y), as well to all vertices in $\rho[(x, y)]$, and colouring all vertices in $\rho[(y, x)]$ by τ. Note that if (x, y) is the lexicographically smallest pair chosen by Step 3 then $\{x\} \ll \{y\}$.

Now we prove the correctness of Algorithm 3.2.4. First we have the following straightforward observation.

Lemma 3.2.5 *Let D be a local tournament. Then D is a local transitive tournament if and only if neither the outset nor the inset of any vertex contains a directed triangle.* □

Let a, b, c, and d be four vertices of D. If $\{b, c, d\}$ induces a directed triangle and a is dominated by $\{b, c, d\}$ or dominates $\{b, c, d\}$, then we call $\{a, b, c, d\}$ a *forbidden quadruplet.*

Suppose that $D = (V, A)$ is an orientation of G obtained by Algorithm 3.2.4. By Theorem 2.1.3, we know that D is a local tournament. Assume that D is not a local transitive tournament, i.e., that D contains a forbidden quadruplet. Among all forbidden quadruplets of D, let $\{a, b, c, d\}$ be the lexicographically smallest one. Assume that a dominates $\{b, c, d\}$ which induces a directed triangle $b \rightarrow c \rightarrow d \rightarrow b$. (A similar argument applies if a is dominated by $\{b, c, d\}$.) Since $\{b, c, d\}$ induces a triangle, there are two vertices $x, y \in \{b, c, d\}$ such that $x \rightarrow y$ and $\{y\} \ll \{x\}$. Without loss of generality, assume that d and b are two such vertices, that is, $\{b\} \ll \{d\}$. Then there exists an ordered pair $(d', b') \in F(G)$, which was the lexicographically smallest one chosen by Step 3 of Algorithm 3.2.4 such that $\rho[(d', b')] = \rho[(d, b)]$. Note that $\{d'\} \ll \{b'\}$ and $\{d', b'\} \ll \{d, b\}$.

Since $\rho[(d', b')] = \rho[(d, b)]$, there exist $(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l) \in F(G)$ such that

$$(d, b) = (x_1, y_1)\Gamma(x_2, y_2)\Gamma \ldots \Gamma(x_l, y_l) = (d', b').$$

Since $(d', b') \neq (d, b)$, $l \geq 2$. Without loss of generality, we assume that $(x_i, y_i) \neq (x_{i+1}, y_{i+1})$ for each $i = 1, 2, \ldots, l-1$. By Lemma 2.1.8, x_i dominates y_i in D for each $i = 1, 2, \ldots, l$.

For convenience, we now change notation. Let $b_i = y_i$ if i is odd and $b_i = x_i$ if i is even. Let $d_i = x_i$ if i is odd and $d_i = y_i$ if i is even. Then $(b_i, d_i)\Gamma(d_{i+1}, b_{i+1})$ for each $i = 1, 2, \ldots, l-1$. Moreover $b_i \rightarrow a_i$ if i is even and $d_i \rightarrow b_i$ if i is odd.

Claim: The following three statements are true

1. For each $i = 1, 2, \ldots, l$, d_i is adjacent to every vertex of $\{a, b, c\}$.

2. For each $i = 1, 2, \ldots, l$, b_i is adjacent to every vertex of $\{a, c, d\}$.

3. There exists a forbidden quadruplet which is lexicographically less than $\{a, b, c, d\}$.

Proof: We apply induction on l. Assume first that $l = 2$. Note that $d_1 = d$ is adjacent to every vertex of $\{a, b, c\}$, and that $b_1 = b$ is adjacent to every vertex of $\{a, c, d\}$. Since $(d_1, b_1)\Gamma(b_2, d_2)$, either $d_1 = d_2$, $b_1 \neq b_2$, and b_1 is not adjacent to b_2, or $b_1 = b_2$, $d_1 \neq d_2$, and d_1 is not adjacent to d_2. Suppose first that $d_1 = d_2$, $b_1 \neq b_2$, and b_1 is not adjacent to b_2. Since $a \rightarrow d_1$, $b_2 \rightarrow d_2$, and $d_1 = d_2$, we have that b_2 is adjacent to a. Then $b_2 \rightarrow a$ because $a \rightarrow b_1$ and b_1 is not adjacent to b_2. Since $c \rightarrow d_1$, $b_2 \rightarrow d_2$, and $d_1 = d_2$, we have that b_2 is adjacent to c. Then $c \rightarrow b_2$ because $b_1 \rightarrow c$ and b_1 is not adjacent to d_2. Statements 1 and 2 now follow. We notice that $\{a, c, b_2\}$ induces a directed triangle which dominates d_2. Then $\{a, c, b_2, d_2\}$ is a forbidden quadruplet in D. This quadruplet is lexicographically less than $\{a, b, c, d\}$ because $\{b_2, d_2\} \ll \{b, d\}$.

Suppose in turn that $b_1 = b_2$, $d_1 \neq d_2$, and d_1 is not adjacent to d_2. Since $b_1 \rightarrow c$, $b_2 \rightarrow d_2$, and $b_1 = b_2$, we have that d_2 is adjacent to c. Then $d_2 \rightarrow c$ because $c \rightarrow d_1$ and d_1 is not adjacent to d_2. Hence d_2 is adjacent to a as $\{a, d_2\} \rightarrow c$. Since $a \rightarrow d_1$ and d_1 is not adjacent to d_2, we must have $d_2 \rightarrow a$. Statements 1 and 2 now follow. We notice that $\{a, b_2, d_2\}$ induces a directed triangle which dominates c. So $\{a, c, b_2, d_2\}$ is a forbidden quadruplet of D. This quadruplet is lexicographically less than $\{a, b, c, d\}$ as $\{b_2, d_2\} \ll \{b, d\}$.

Now we assume that the Claim is true for all $l \leq k$ and we consider the case when $l = k + 1$ where $k \geq 2$.

Let $B_k = \{b_1, b_2, \ldots, b_k\}$ and $D_k = \{d_1, d_2, \ldots, d_k\}$. By the induction hypothesis, every vertex of $\{a, b, c\}$ is adjacent to every vertex of D_k, and every vertex of $\{a, c, d\}$

is adjacent to every vertex of B_k. Let $b_j, b_{j+1} \in B_k$. Suppose that $b_j \neq b_{j+1}$. Then b_j and b_{j+1} are not adjacent. So if $a \rightarrow b_j$ (resp. $b_j \rightarrow a$) then $b_{j+1} \rightarrow a$ (resp. $a \rightarrow b_{j+1}$). We know that $a \rightarrow b_1$. Hence

$a \rightarrow b_k$ if $|B_k|$ is odd and $b_k \rightarrow a$ if $|B_k|$ is even.

Applying a similar argument, we can show

$a \rightarrow d_k$ if $|D_k|$ is odd and $d_k \rightarrow a$ if $|D_k|$ is even,

$b \rightarrow d_k$ if $|D_k|$ is even and $d_k \rightarrow b$ if $|D_k|$ is odd,

$c \rightarrow b_k$ if $|B_k|$ is even and $b_k \rightarrow c$ if $|B_k|$ is odd,

$c \rightarrow d_k$ if $|D_k|$ is odd and $d_k \rightarrow c$ if $|D_k|$ is even, and

$d \rightarrow b_k$ if $|B_k|$ is odd and $b_k \rightarrow d$ if $|B_k|$ is even.

Since $(b_k, d_k)\Gamma(d_{k+1}, b_{k+1})$, either $b_k = b_{k+1}$, $d_k \neq d_{k+1}$, and d_k is not adjacent to d_{k+1}, or $d_k = d_{k+1}$, $b_k \neq b_{k+1}$, and b_k is not adjacent to b_{k+1}.

Since the two cases are similar, we only consider the case when $b_k = b_{k+1}$, $d_k \neq d_{k+1}$, and d_k is not adjacent to d_{k+1}. An important fact to observe is that the integers k and $|B_k| + |D_k|$ have distinct parity. We discuss the following cases.

Case 1. Suppose that k is odd. In this case, $|B_k|$ and $|D_k|$ have the same parity.

Subcase 1.1. Suppose that both $|B_k|$ and $|D_k|$ are odd. Then from the above discussion, we know that $a \rightarrow b_k$, $a \rightarrow d_k$, $d_k \rightarrow b$, $c \rightarrow d_k$, $b_k \rightarrow c$, and $d \rightarrow b_k$. Since k is odd, we have $b_{k+1} \rightarrow d_{k+1}$. Thus d_{k+1} is adjacent to c because $b_k \rightarrow c$, $b_{k+1} \rightarrow d_{k+1}$, and $b_k = b_{k+1}$. Hence $d_{k+1} \rightarrow c$ as $c \rightarrow d_k$ and d_k is not adjacent to d_{k+1}. We see that d_{k+1} is adjacent to a and b because $\{a, b, d_{k+1}\} \rightarrow c$. Since $a \rightarrow d_k$ and d_k is not adjacent to d_{k+1}, we have $d_{k+1} \rightarrow a$. Statements 1 and 2 now follow easily. We notice that $\{a, b_{k+1}, d_{k+1}\}$ induces a directed triangle which dominates c. Then $\{a, c, b_{k+1}, d_{k+1}\}$ is a forbidden quadruplet of D. This quadruplet is lexicographically less than $\{a, b, c, d\}$ since $\{b_{k+1}, d_{k+1}\} \ll \{b, d\}$.

Subcase 1.2. Suppose that both $|B_k|$ and $|D_k|$ are even. Then $b_k \rightarrow a$, $d_k \rightarrow a$, $b \rightarrow d_k$, $c \rightarrow b_k$, $d_k \rightarrow c$, and $b_k \rightarrow d$. Since $b_{k+1} \rightarrow d_{k+1}$, $b_k \rightarrow a$, and $b_k = b_{k+1}$, we have that d_{k+1} is adjacent to a. Then $a \rightarrow d_{k+1}$ because $d_k \rightarrow a$ and d_k is not adjacent to d_{k+1}. Now we have $a \rightarrow \{b, c, d_{k+1}\}$ and hence d_{k+1} is adjacent to b and c. Thus statements 1 and 2 have been proved. Since $b \rightarrow d_k$ and d_k is not adjacent to d_{k+1}, we have $d_{k+1} \rightarrow b$. Since $d_k \rightarrow c$ and d_k is not adjacent to d_{k+1}, we have $c \rightarrow d_{k+1}$. Thus $\{a, c, b_{k+1}\}$ induces a directed triangle which dominates d_{k+1}. So $\{a, c, b_{k+1}, d_{k+1}\}$ is a forbidden quadruplet of D which is lexicographically less than $\{a, b, c, d\}$ because $\{b_{k+1}, d_{k+1}\} \ll \{b, d\}$.

Case 2. Suppose that k is even. Then $|B_k|$ and $|D_k|$ have distinct parity.

Subcase 2.1. Suppose that $|B_k|$ is odd and that $|D_k|$ is even. Then $a \rightarrow b_k$, $d_k \rightarrow a$, $b \rightarrow d_k$, $b_k \rightarrow c$, $d_k \rightarrow c$, and $d \rightarrow b_k$. We note that $d_{k+1} \rightarrow b_{k+1}$ since k is even. Then d_{k+1} is adjacent to a because $d_{k+1} \rightarrow b_{k+1}$, $a \rightarrow b_k$, and $b_k = b_{k+1}$. Since $d_k \rightarrow a$ and d_k is not adjacent to d_{k+1}, we have $a \rightarrow d_{k+1}$. Thus $a \rightarrow \{b, c, d_{k+1}\}$ and hence d_{k+1} is adjacent to b and c. Since $b \rightarrow d_k$ and d_k is not adjacent to d_{k+1}, we have $d_{k+1} \rightarrow b$. Since $d_k \rightarrow c$ and d_k is not adjacent to d_{k+1}, we have $c \rightarrow d_{k+1}$. Hence statements 1 and 2 have been proved. We see that $\{c, b_{k+1}, d_{k+1}\}$ induces a directed triangle which is dominated by a. So $\{a, c, b_{k+1}, d_{k+1}\}$ is a forbidden quadruplet of D which is lexicographically less than $\{a, b, c, d\}$ because $\{b_{k+1}, d_{k+1}\} \ll \{b, d\}$.

Subcase 2.2. Suppose that $|B_k|$ is even and that $|D_k|$ is odd. Then $b_k \rightarrow a$, $a \rightarrow d_k$, $d_k \rightarrow b$, $c \rightarrow b_k$, $c \rightarrow d_k$, and $b_k \rightarrow d$. Since $d_{k+1} \rightarrow b_{k+1}$, $c \rightarrow b_k$, and $b_k = b_{k+1}$, we have that d_{k+1} is adjacent to c. Then $d_{k+1} \rightarrow c$ because $c \rightarrow d_k$ and d_k is not adjacent to d_{k+1}. Thus we have $\{a, b, d_{k+1}\} \rightarrow c$ and hence d_{k+1} is adjacent to a and b. Hence statements 1 and 2 follow. Since $a \rightarrow d_k$ and d_k is not adjacent to d_{k+1}, we have $d_{k+1} \rightarrow a$. Since $d_k \rightarrow b$ and d_k is not adjacent to d_{k+1}, we have $b \rightarrow d_{k+1}$. Now we see that $\{a, c, b_{k+1}\}$ induces a directed triangle which is dominated by d_{k+1}. So $\{a, c, b_{k+1}, d_{k+1}\}$ is a forbidden quadruplet of D which is lexicographically less than $\{a, b, c, d\}$ as $\{b_{k+1} d_{k+1}\} \ll \{b, d\}$. □

However statement 3 of the above claim contradicts our choice of $\{a, b, c, d\}$. Therefore D contains no forbidden quadruplet and D is a local transitive tournament by Lemma 3.2.5. In conjunction with Theorem 2.1.15 and Corollary 2.2.10, we have proved the following result.

Theorem 3.2.6 *Algorithm 3.2.4 finds a local-transitive-tournament orientation of G if one exists, and otherwise correctly reports that one does not exist.* □

We now present a simple proof of Skrien's Theorem which states that a connected graph is local-tournament-orientable if and only if it is a proper circular arc graph, as an application of our lexicographic method.

Proof of Theorem 2.1.1: Suppose that G is a proper circular arc graph with a circular arc representation F. By Lemma 2.2.8 the representation $\mathcal{F}$ can be chosen so that no two arcs share a common endpoint and no two arcs together cover the entire circle. Let $S_1, S_2, \ldots, S_n$ be the arcs in $\mathcal{F}$. For each $i = 1, 2, \ldots, n$, let v_i be the vertex of G associated with S_i. We obtain an oriented graph D as follows: The vertex set of D is the same as the vertex set of G, and a vertex v_i dominates a vertex v_j in D if and only if S_i contains the head of S_j (or S_j contains the tail of S_i). By Theorem 2.2.6, D is a local-transitive-tournament orientation of G. Hence G is local-tournament-orientable.

Suppose in turn that G is local-tournament-orientable. Then by Theorem 3.2.6, G admits a local-transitive-tournament orientation. Hence by Theorem 2.2.6, G is a proper circular arc graph. □

3.2.3 The Acyclic Local-tournament Orientation Algorithm for Proper Interval Graphs

A closed walk $C = v_1 \sim v_2 \sim \ldots \sim v_k \sim v_1$ is called a *semicycle* if v_{i-1} is not adjacent to v_{i+1} for each $i = 1, 2, \ldots, k$, where the subscript addition and subtraction are modulo k. The following lemma is the traditional characterization of interval graphs due to Gilmore and Hoffman (cf. [32]).

Lemma 3.2.7 *A graph G is an interval graph if and only if it contains no chordless cycle of length 4 and $\overline{G}$ contains no semicycles of odd length.* □

A proper interval graph is of course an interval graph. However the converse is not necessarily true, that is, not all interval graphs are proper interval graphs. The following result, due to Roberts [68], shows which interval graphs are proper interval graphs.

Lemma 3.2.8 *An interval graph is a proper interval graph if and only if it is claw-free.* □

Theorem 3.2.9 *The following statements are equivalent for a graph G.*

1. *G is a proper interval graph,*

2. *G is orientable as a non-strong local tournament,*

3. *G is orientable as an acyclic local tournament,*

4. *G is claw-free, contains no chordless cycle of length 4, and $\overline{G}$ contains no semi-cycles of odd length.*

Proof: It suffices to show that the statements of Theorem 3.2.9 are equivalent for a connected graph G. The equivalence between statement 1 and statement 4 is implied by Lemma 3.2.7 and Lemma 3.2.8.

$1 \Rightarrow 2$: Assume that G is a proper interval graph and assume that $\mathcal{I}$ is a proper interval representation of G. Without loss of generality, assume that the intervals of $\mathcal{I}$ have distinct endpoints. Let $I_1, I_2, \ldots, I_n$ be the intervals of $\mathcal{I}$ and let v_i be the vertex of G associated with I_i for each $i = 1, 2, \ldots, n$. We obtain an oriented graph D as follows: Let the vertex set of D be the vertex set of G and let $v_i \rightarrow v_j$ if S_i contains the left endpoint of S_j. We note that S_i contains the left endpoint of S_j if and only if S_j contains the right endpoint of S_i. Moreover, for any two intersecting intervals S_i and S_j, either S_i contains the left endpoint of S_j or S_j contains the left endpoint of S_i. Thus each edge of G is assigned exactly one direction and so D is an orientation of G. Since no interval of $\mathcal{I}$ contains the left endpoint of the left-most interval of $\mathcal{I}$, the corresponding vertex associated with the first interval of $\mathcal{I}$ has no in-neighbour in D. Hence D is non-strong. For each vertex v_i, the out-neighbours of v_i are associated with those intervals of $\mathcal{I}$ containing the right endpoint of S_i. Hence the intervals associated with the out-neighbours of v_i intersect each other. So the out-neighbours of v_i induce a complete subgraph of D. A similar discussion applies to show that the in-neighbours of v_i induce a complete subgraph of D. Therefore D is a local tournament.

$2 \Rightarrow 3$: Suppose that G is orientable as a non-strong local tournament. Then by Corollary 2.2.5, G is straight-orientable. Since a straight oriented graph is an acyclic local tournament, G is orientable as an acyclic local tournament.

$3 \Rightarrow 1$: Suppose that G is orientable as an acyclic local tournament. Then by Corollary 2.2.5, G admits a straight orientation. Hence G is a proper interval graph by Theorem 2.2.2. □

A graph G admits a *perfect elimination order* if the vertices of G can be linearly ordered $v_1, v_2, \ldots, v_n$ so that for each v_i the vertices adjacent to v_i with subscripts greater than i induce a complete subgraph. It has been proved by Fulkerson and Gross that a graph admits a perfect elimination order if and only if it is chordal (cf. [24]).

Suppose that G is a proper interval graph. Then G is a proper circular arc graph and hence G^* is 2-colourable. Moreover G must be chordal and therefore it admits a perfect elimination order. Given a perfect elimination order, $v_1, v_2, \ldots, v_n$, we define the lexicographic order according to this order.

Consider the following algorithm for finding acyclic local-tournament orientations.

Algorithm 3.2.10 *Let G be a graph.*

Step 1. Construct the characteristic graph G^* of G.

Step 2. If G^* is not 2-colourable or G does not admit a perfect elimination order, then G is not a proper interval graph.

Step 3. Find a perfect elimination order of G, $v_1, v_2, \ldots, v_n$.

Step 4. Find a special 2-colouring of G^* with colours μ and τ by always first assigning colour μ to the lexicographically (with respect to the perfect elimination order) smallest uncoloured vertex (x, y), and then complete the unique 2-colouring of the corresponding component of G^*.

Step 5. Let A be the set of vertices of G^* which are coloured with μ and obtain an acyclic local-tournament orientation $D = (V, A)$ of G. □

The following lemma is a consequence of Theorem 2.3.1 and Theorem 3.2.9.

Lemma 3.2.11 *Let G be a proper interval graph. Then G can be obtained from a reduced straight-orientable graph S by substituting a complete graph for each vertex of S.* □

Lemma 3.2.12 *Let S be a reduced connected graph with a straight enumeration $x_1, x_2, \ldots, x_l$. Suppose that $G = (V, E)$ is a graph obtained from S by substituting a complete graph H_{x_i} for each vertex x_i of S. If $\Delta(G) < n - 1$, then the following hold:*

1. *For each x_i, each edge of H_{x_i} forms one implication class, that is, if $u, v \in H_{x_i}$ and $(u, v)\Gamma^*(u', v')$ for some (u', v') then $(u, v) = (u', v')$.*
2. *All edges of G which are not in H_{x_i} for any x_i form one implication class, that is, if $(z_i, z_j), (z_a, z_b) \in F(G)$ where $z_i \in H_{x_i}$, $z_a \in H_{x_a}$, $z_j \in H_{x_j}$, and $z_b \in H_{x_b}$ with $i < j$ and $a < b$, then $(z_i, z_j)\Gamma^*(z_a, z_b)$.*

Proof: The first assertion of the lemma is easy. To prove the second assertion, we first note that S is uniquely orientable as a non-strong local tournament. In fact, if $(x_i, x_j), (x_a, x_b) \in F(S)$ where $i < j$ and $a < b$, then $(x_i, x_j)\Gamma^*(x_a, x_b)$. Suppose that $y_1, y_2, \ldots, y_l$ are vertices of G such that $y_i \in H_{x_i}$ for each $i = 1, 2, \ldots, l$. Then $< \{y_1, y_2, \ldots, y_l\} >$ is a subgraph of G which is just a copy of S. If $(y_i, y_j), (y_a, y_b) \in F(G)$ where $i < j$ and $a < b$, then $(y_i, y_j)\Gamma^*(y_a, y_b)$. The rest of the proof follows b Lemma 2.3.14. □

Lemma 3.2.13 *Let S be a reduced connected graph with a straight enumeration $x_1, x_2, \ldots, x_l$. Suppose that $G = (V, E)$ is a graph which is obtained from S by substituting a complete graph H_{x_i} for each vertex x_i of S. If $\Delta(G) = n - 1$, then:*

1. *l is odd, and H_{x_r}, with $r = \frac{l+1}{2}$, consists of vertices of G of degree $n-1$;*

2. *For each x_i, each edge of H_{x_i} forms one implication class, that is, if $u, v \in H_{x_i}$ and $(u,v)\Gamma^*(u',v')$ for some (u',v') then $(u,v) = (u',v')$;*

3. *For each $u \in H_{x_r}$ all edges uv with $v \notin H_{x_r}$ form one implication class, that is, if $z_i \in H_{x_i}$, $z_j \in H_{x_j}$; $z_a \in H_{x_a}$, and $z_b \in H_{x_b}$ with $i, j < r < a, b$, then $(u, z_i)\Gamma^*(u, z_j)$, $(u, z_a)\Gamma^*(u, z_b)$, and $(u, z_i)\Gamma^*(z_a, u)$;*

4. *All edges not in H_{x_i} for any x_i and not incident with any vertex of H_{x_r} form one implication class, that is, if $(z_i, z_j), (z_a, z_b) \in F(G)$ where $z_i \in H_{x_i}$, $z_j \in H_{x_j}$, $z_a \in H_{x_a}$, and $z_b \in H_{x_b}$ with $i, j, a, b \neq r$, $i < j$, and $a < b$, then $(z_i, z_j)\Gamma^*(z_a, z_b)$.*

Proof: Assertion 1 is a consequence of Proposition 2.3.5. Assertion 2 is easy. To prove assertions 3 and 4, let $y_1, y_2, \ldots, y_l$ be a set of vertices of G such $y_i \in H_{x_i}$ for each $i = 1, 2, \ldots, l$. Then $< \{y_1, y_2, \ldots, y_l\} >$ is a reduced connected graph which is a copy of S. Hence it can be oriented as a non-strong local tournament. By Proposition 2.3.5, $< \{y_1, y_2, \ldots, y_l\} >$ has precisely two implication classes: One class consists of all edges that are incident with y_r ($r = \frac{l+1}{2}$), the other class consists of all edges that are not incident with y_r. In fact, if $i, j < r < a, b$, then $(y_r, y_i)\Gamma^*(y_r, y_j)$, $(y_r, y_a)\Gamma^*(y_r, y_b)$, and $(y_r, y_i)\Gamma^*(y_a, y_r)$. Moreover, if $(y_i, y_j), (y_a, y_b) \in F(G)$ where $i, j, a, b \neq r$, $i < j$, and $a < b$, then $(y_i, y_j)\Gamma^*(y_a, y_b)$. The rest of the proof follows by using Lemma 2.3.14. □

If G is a proper interval graph, then G is a proper circular arc graph and hence G^* is 2-colourable. Recall from Chapter 2 that if $(u,v), (x,y) \in F(G)$ with $(u,v)\Gamma^*(x,y)$ then (u,v) and (x,y) must be coloured with the same colour in any 2-colouring of G^*.

Theorem 3.2.14 *Algorithm 3.2.10 correctly produces an acyclic local-tournament orientation $D = (V, A)$ of G, provided one exists, and otherwise correctly reports that one does not exist.*

Proof: Clearly the algorithm finds that an acyclic local-tournament orientation does not exist if and only if this is the case (cf. Theorem 3.2.9). Suppose that

$G = (V, E)$ is a proper interval graph and $D = (V, A)$ is the orientation of G obtained by Algorithm 3.2.10. Let S be a reduced connected straight-orientable graph with a straight enumeration $x_1, x_2, \ldots, x_l$. Suppose that G is a graph which is obtained from S by substituting a complete graph H_{x_i} for each vertex x_i of S.

Let $v_1, v_2, \ldots, v_n$ be a perfect elimination order of G. Consider H_{x_i} for some x_i. Assume that $b_1, b_2, \ldots, b_p$ are vertices of H_{x_i} listed in the perfect elimination order of G. Note that each $\{(b_j, b_k)\}$ is a Γ^*-class. According to Step 4 the colour μ is always assigned to (b_j, b_k) with $j < k$. Hence $\{(b_j, b_k) | \ j < k\} \subseteq A$. Therefore each H_{x_i} obtains a transitive tournament orientation.

We note that Algorithm 3.2.10 is the same as Algorithm 3.2.4 except that it chooses a special order, namely, a perfect elimination order of vertices of G. Then by Theorem 3.2.6 D is a local-transitive-tournament orientation of G. So it suffices to show that D is acyclic.

We consider the following two cases.

Case 1. Suppose that $\Delta(G) \leq n - 2$. Then all edges of G which are not in H_{x_i} for any x_i form an implication class. Without loss of generality, assume that $v_1 \in H_{x_f}$ and assume that v_h is the first vertex in the perfect elimination order which is adjacent to v_1 and is not in H_{x_f}. Suppose that v_h is in H_{x_g}, where $g \neq h$. Without loss of generality, assume that $g > h$. Then (v_1, v_h) is coloured with colour μ according to the Step 4 of Algorithm 3.2.10. By Lemma 3.2.12, if $x \in H_{x_i}$ and $y \in H_{x_j}$, with $i < j$, and if $(x, y) \in F(G)$, then (x, y) must be coloured with μ. We know that each H_{x_i} is oriented as a transitive tournament. Therefore D may be viewed as an oriented graph obtained from a straight orientation of S by substituting a transitive tournament for each vertex of S. Hence D is acyclic.

Case 2. Suppose that $\Delta(G) = n - 1$. By Lemma 3.2.13, l is odd and H_{x_r} consists of vertices of degree $n - 1$, where $r = \frac{l+1}{2}$. We consider the first vertex v_1 in the perfect elimination order. Suppose that $v_1 \in H_{x_i}$. If $1 < i < l$, let $x_i x_a$ and $x_i x_b$ be the left-most wave and the right-most wave at x_i in the straight enumeration of S, then x_a is not adjacent to x_b. Then v_1 has two non-adjacent neighbours, violating the perfect elimination order. Thus $i = 1$, or l.

Assume without loss of generality that $i = 1$. Suppose that $a_1, a_2, \ldots, a_q$ are the vertices of H_{x_r}. Then $(v_1, a_1), (v_1, a_2), \ldots, (v_1, a_q)$ are chosen by Step 4 of Algorithm 3.2.10 to be lexicographically smallest. Hence they are coloured with μ. Therefore, by Lemma 3.2.13, if $u \in H_{x_i}$ with $i < r$ then (u, a_j) is coloured with μ, and if $v \in H_{x_k}$ with $k > r$, then (a_j, v) is coloured with μ.

Suppose that there is a vertex which is adjacent to v_1 but not in H_{x_1} (note that such a vertex may not exist when $l = 3$, if there is no such vertex then D is easily seen to be acyclic). Let v_k be the first such vertex in the perfect elimination order. Then (v_1, v_k) is chosen by Step 4 of Algorithm 3.2.10 to colour with μ. Thus, by Lemma 3.2.13, if $u \in H_{x_i}$ and $v \in H_{x_j}$ with $i < j$ and $i, j \neq r$, and if $(x, y) \in F(G)$, then (u, v) must be coloured with μ.

As we have shown above, each H_{x_i} obtains a transitive tournament orientation. So D may be viewed as an oriented graph obtained from a straight orientation of S by substituting a transitive tournament for each vertex of S. Therefore D is acyclic. □

From the above proof we see that Step 2 is not necessary if $\Delta(G) < n - 1$.

Chapter 4

Recognition and Representation Algorithms

4.1 Introduction

The algorithmic aspects of interval graphs have been extensively studied, [33] – in particular, the recognition and the representation problems for interval graphs have been solved by Booth and Lueker [17] with an $O(m+n)$ algorithm. The algorithm given by Booth and Lueker relies on a complicated data structure called a PQ-tree. Another $O(m+n)$ time algorithm for solving the same problem was later obtained by Korte and Möhring [53]. Again the algorithm uses PQ-trees. Since then, many people tried to find a simpler algorithm without using PQ-trees. For proper interval graphs, we solve the problem with an $O(m+n)$ algorithm. Our algorithm makes use of our structure theorem for proper interval graphs instead of PQ-trees. Recently Hsu [44] announced a simple $O(m+n)$ algorithm for testing interval graphs without using PQ-trees.

It is a longstanding open problem to find an $O(m+n)$ time algorithm for the recognition and representation of circular arc graphs. However, for *proper* circular arc graphs, we have mentioned that Tucker gave a matrix characterization, and a recognition algorithm of complexity $O(n^2)$. In Section 3.2 we gave an $O(m\Delta)$ time algorithm to recognize proper circular arc graphs and to obtain local-transitive-tournament orientations. As we have seen in Section 2.2, a local-transitive-tournament orientation is equivalent to a proper circular arc representation, hence the representation problem for proper circular arc graphs can also be solved in $O(m\Delta)$ time. In Section 4.3, we will give the first optimal algorithms, i.e., of complexity $O(m+n)$, for the recognition and representation of proper circular arc graphs.

A *mixed graph* has some directed edges (i.e., arcs) and some undirected edges. The terms, 'block', 'inset', 'outset', in a mixed graph can be defined in a similar fashion as in a digraph (cf. Section 1.2). For instance, two vertices are in the same block if and only if they have the same closed neighbourhood in the underlying graph.

We shall be dealing with a particular kind of mixed graph. Let $V_1, V_2, \ldots, V_p$ be the blocks of H. Then H is a *mixed local tournament* provided all edges of H within each block V_i are undirected, all edges of H between two fixed blocks V_i and V_j are directed in the same direction (all from V_i to V_j or all from V_j to V_i), and provided the inset as well as the outset of every vertex is a complete mixed graph, i.e., any two vertices are adjacent by some (directed or undirected) edge. A mixed local tournament is *acyclic* if it contains no directed cycle.

Note that each block is a complete subgraph. So if H is a mixed local tournament, then a local tournament can easily be obtained from H by assigning any orientation to each block V_i. If in addition H is an acyclic mixed local tournament, then an acyclic local tournament can be obtained from H by assigning a transitive orientation to each V_i.

Suppose that H is a mixed local tournament. If we reverse all arcs in H, then we again get a mixed local tournament. We call the operation of reversing all arcs (directed edges) in a mixed graph also a *full reversal.* (It extends the earlier definition we gave for a full reversal in an oriented graph.) In particular, if H is an acyclic mixed local tournament, then by the full reversal of H we again obtain an acyclic mixed local tournament.

Let H be a mixed local tournament. Suppose that S is a subgraph of H which contains one vertex from each block of H. It is clear that S is a local tournament.

If in addition H is acyclic, then S is an acyclic local tournament and hence admits a straight enumeration. So if H is an acyclic mixed local tournament, then the blocks can be uniquely ordered $V_1, V_2, \ldots, V_p$ so that $V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow V_p$, and, for each $x \in V_j$, there exist l_j and r_j (which depend on j) such that

$$I(x) = V_{j-1} \cup V_{j-2}, \ldots, \cup V_{j-l_j}; \; O(x) = V_{j+1} \cup V_{j+2}, \ldots, \cup V_{j+r_j}.$$

We call this order of blocks the *straight enumeration.* We call $V_jV_{j-l_j}$ the *left-most wave* and $V_jV_{j+r_j}$ the *right-most wave* at the block V_j. An acyclic mixed local tournament is also called a *straight mixed graph.*

The straight enumeration of the blocks of a straight mixed graph is very similar to the straight enumeration of the vertices of a reduced straight oriented graph. They share many properties. For instance, if V_i and V_j ($i < j$) are adjacent blocks, then $V_i \cup V_{i+1} \cup \ldots \cup V_j$ is complete.

If H is a mixed graph obtained from a graph G by assigning directions to edges of G, then H is a *mixed-graph orientation* of G. If in addition H is a straight mixed graph, then G is *orientable as a straight mixed graph* or *straight-mixed-graph-orientable*, and H is a *straight-mixed-graph orientation* of G. If G has precisely two straight-mixed-graph orientations for which each is obtained from the other by full reversal, then G is *uniquely orientable as a straight mixed graph.*

For proper interval graphs the situation is very simple.

Theorem 4.1.1 *A connected proper interval graph G is uniquely orientable as a straight mixed graph.*

Proof: By Theorem 2.3.1. □

4.2 Proper Interval Graphs

In this section, we give an $O(m + n)$ time algorithm to recognize proper interval graphs. Assume that G is a connected graph, as otherwise we can work separately on each component of G. Our algorithm will insert vertices of G one by one into an already formed straight mixed graph to form a new straight mixed graph. If G is a proper interval graph, then, as we shall show below, this process continues until a straight-mixed-graph orientation of G is obtained. Since a straight-mixed-graph orientation of G can be easily modified to an acyclic local-tournament orientation of G, G can be represented by a proper interval family (see Theorem 2.2.2).

By Theorem 4.1.1, a straight-mixed-graph orientation of a connected proper interval graph is unique. So the corresponding straight enumeration of blocks is unique. This is crucial in what follows, even though it is not always explicitly mentioned.

We state our algorithm as follows.

Algorithm 4.2.1 *Let $G = (V, E)$ be a connected graph.*

[Step 1.] Order the vertices of G as $v_1, v_2, \ldots, v_n$ in such a way that $< \{v_1, v_2, \ldots, v_i\} >$ is connected, for each $i = 1, 2, \ldots, n$.

[Step 2.] Let $H_1 =< \{v_1\} >$ and $i = 1$. While possible, insert v_{i+1} into H_i to form a straight mixed graph H_{i+1}, and increase i by 1. □

For Step 1, we may apply breadth-first search to sort the vertices as required. This can be done in time $O(m + n)$. Moreover, we may arrange to store, for each i, a vertex $v_j, j < i$, such that v_i is adjacent to v_j. For Step 2, suppose that G is a proper interval graph and suppose that, for some $i = 1, 2, \ldots, n-1$, $< \{v_1, v_2, \ldots, v_i\} >$ has been oriented as a straight mixed graph H_i. Then the orientation H_i is unique.

Note that $< \{v_1, v_2, \ldots, v_i, v_{i+1}\} >$ is also uniquely orientable as a straight mixed graph. If H_{i+1} is a straight-mixed-graph orientation of $< \{v_1, v_2, \ldots, v_i, v_{i+1}\} >$, then H_{i+1} agrees with the orientation on H_i, up to full reversal. Using a similar approach, we conclude that any straight-mixed-graph orientation D of G agrees with the orientation on H_i, up to full reversal. Therefore, to obtain a straight-mixed-graph orientation of $< \{v_1, v_2, \ldots, v_i, v_{i+1}\} >$, we need only add v_{i+1} to H_i and appropriately assign directions to some undirected edges.

Let $V_1, V_2, \ldots, V_p$ be the straight enumeration of the blocks of H_i, and let D be a straight-mixed-graph orientation of G which agrees with the orientation on H_i.

Fact 1. Suppose that V_a, V_b, and V_c, where $a < b < c$, are three blocks of H_i. If v_{i+1} is adjacent to a vertex in V_a and to a vertex in V_c, then v_{i+1} is adjacent to every vertex in V_b.

Proof of Fact 1: Assume that v_{i+1} is adjacent to $x \in V_a$ and $z \in V_c$ but not to $y \in V_b$. Choose such a, b, and c with $c - a$ minimal. Then $c - a \geq 2$ and v_{i+1} is not adjacent to at least one vertex in V_d for each d such that $a < d < c$. Since v_{i+1} is not adjacent to a vertex in V_{a+1} and $V_a \rightarrow V_{a+1}$, v_{i+1} must dominate x in D. Similarly since $V_{c-1} \rightarrow V_c$ and v_{i+1} is not adjacent to a vertex in V_{c-1}, v_{i+1} must be dominated by z in D. Hence there is a directed cycle of D contained in

$$v_{i+1} \rightarrow x \rightarrow V_{a+1} \rightarrow \ldots \rightarrow V_{c-1} \rightarrow z \rightarrow v_{i+1},$$

contradicting the fact that D is acyclic. □

Fact 2. Let V_a, V_b, and V_c, where $a < b < c$, be three blocks of H_i. Suppose that v_{i+1} is adjacent to $y \in V_b$ and suppose that v_{i+1} is not adjacent to $x \in V_a$ and $z \in V_c$. Then V_a is completely non-adjacent to V_c.

Proof of Fact 2: Assume that V_a is completely adjacent to V_c. The blocks V_a and V_b must have distinct closed neighbourhoods, i.e., there is a block which is completely adjacent to exactly one of V_a and V_b. If there is a block V_d which is completely adjacent to V_b but non-adjacent to V_a, then $d > c$ because V_a is completely adjacent to V_c. Note that v_{i+1} is not adjacent to any vertex in V_d according to Fact 1. Hence for any $w \in V_d$ $\{x, y, w, v_{i+1}\}$ induces a copy of the claw in G, contradicting the fact that G is local-tournament-orientable. Thus there must be a block V_e which is completely adjacent to V_a but non-adjacent to V_b. Similarly there is a block V_f which is completely adjacent to V_c but non-adjacent to V_b. Note that $e < a$ and $f > c$. Hence v_{i+1} is adjacent to no vertex in V_e or V_f. Therefore, for any $u \in V_e$ and $v \in V_f$, $\{x, y, z, u, v, v_{i+1}\}$ induces a copy of the net in G, contradicting the fact that G is local-tournament-orientable. □

Fact 3. Let V_a, V_b, and V_c, where $a < b < c$, be three blocks of H_i. Suppose that $V_a \to V_b \to V_c$. If v_{i+1} is adjacent to some vertex in V_b, then v_{i+1} is adjacent to either every vertex in V_a or every vertex in V_c.

Proof of Fact 3: Suppose that there are three vertices $x \in V_a$, $y \in V_b$, and $z \in V_c$ such that v_{i+1} is adjacent to y but not to x or z. By Fact 2, x is not adjacent to z. Then G contains a copy of the claw induced by $\{x, y, z, v_{i+1}\}$. □

We now insert v_{i+1} into H_i and find a straight mixed graph H_{i+1} which agrees with the orientation on H_i. We discuss the following cases and in each case we express H_{i+1} by the straight enumeration of its blocks.

Case 1. When $p = 1$, i.e., when H_i has only one block V_1, then if v_{i+1} is adjacent to all vertices of V_1 we include v_{i+1} in V_1 and H_{i+1} again has only one block, namely, $V_1 \cup \{v_{i+1}\}$. If there is some S such that $\emptyset \neq S \subset V_1$ and v_{i+1} is adjacent to all vertices of S but to no vertices of $V_1 - S$, then the straight enumeration of the blocks of H_{i+1} is

$$\{v_{i+1}\}, S, V_1 - S.$$

Case 2. When $p \geq 2$, i.e., when H_i has at least two blocks, then according to Fact 1 we may assume that there exist $a < c$ such that v_{i+1} is adjacent to all vertices of each V_j such that $a < j < c$ (if any), and v_{i+1} is not adjacent to any vertex in each V_k such that $k < a$ or $k > c$.

Subcase 2.1. Suppose that v_{i+1} is not adjacent to any vertex in $V_a \cup V_c$.

In this case we must have $c \neq a + 1$ as otherwise v_{i+1} is adjacent to no vertex of H_i, contradicting our hypothesis. Thus $c \geq a + 2$, that is, there is at least one block between V_a and V_c. By Fact 2, V_a is completely non-adjacent to V_c.

Let $V_a V_b$ be the right-most wave at V_a and let $V_c V_d$ be the left-most wave at V_c. Since V_a is not completely adjacent to V_c, blocks V_b and V_d are between V_a and V_c, i.e., $a < b, d < c$. By Fact 3 we must have $b < d$. We claim that $d \leq b + 4$. First we note that for each block V_j with $b < j < d$, V_j is completely adjacent either to V_{a+1} or to V_{c-1}, as otherwise any choice of three vertices from V_{a+1}, V_{c-1}, V_j, respectively, together with v_{i+1} would induce a copy of the claw, contradicting the fact that G is local-tournament-orientable. Now since distinct blocks must have distinct closed neighbourhoods, there are at most three blocks between V_b and V_d.

Suppose that there is no block between V_b and V_d, namely, $d = b + 1$. We prove that

$$\ldots, V_a, \ldots, V_b, \{v_{i+1}\}, V_d, \ldots, V_c, \ldots$$

is the straight enumeration of the blocks of H_{i+1}. To do this, we need to verify that each of the sets above is a block and that any two completely adjacent sets must be adjacent to each set between them.

We need to show that two vertices are in the same set if and only if they have the same closed neighbourhood in H_{i+1}. First it is clear that vertices in each set have the same closed neighbourhoods. Suppose that x and y are two vertices in different sets. If neither x nor y is the vertex v_{i+1}, then x and y have distinct neighbourhoods in H_i and hence in H_{i+1}. Suppose that one of x and y, say x, is the vertex v_{i+1} and suppose that $y \in V_j$ for some j. If $j \leq a$ or $j \geq c$, then x and y are not adjacent and hence have distinct closed neighbourhoods. If $a < j \leq b$, then y is adjacent to the vertices of V_a which are not adjacent to x. If $d \leq j < c$, then y is adjacent to the vertices of V_c which are not adjacent to x. Hence x and y have distinct closed neighbourhoods. Now we shall show that two completely adjacent blocks must be completely adjacent to each block between them. Let A and B be two completely adjacent blocks. Suppose that one of A and B, say A, is the block $\{v_{i+1}\}$ and suppose that $B = V_j$ for some j. Then $a < j < c$ and it is clear that A is completely adjacent to each block between A and B. Suppose that neither A nor B is $\{v_{i+1}\}$. We only need to show that A and B are completely adjacent to $\{v_{i+1}\}$ if $\{v_{i+1}\}$ is between A and B. In fact if $\{v_{i+1}\}$ is a block between A and B, then A and B must be blocks between V_a and V_c and hence A and B must be completely adjacent to $\{v_{i+1}\}$.

In the cases below, similar arguments can be applied to verify that we have defined a straight enumeration. We shall omit the details.

Suppose that $d = b + 2$ and suppose that V_j is the only block between V_b and V_d. If V_j is completely adjacent to V_{a+1} and V_{c-1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j \cup \{v_{i+1}\}, V_d, \ldots, V_c, \ldots$$

If V_j is completely adjacent to V_{a+1} but non-adjacent to V_{c-1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j, \{v_{i+1}\}, V_d, \ldots, V_c, \ldots$$

If V_j is completely adjacent to V_{c-1} but non-adjacent to V_{a+1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, \{v_{i+1}\}, V_j, V_d, \ldots, V_c, \ldots$$

Suppose that $d = b + 3$ and suppose that V_j and V_k are the two blocks between V_b and V_d where $j = b + 1$ and $k = b + 2$. If V_j is completely adjacent to V_{a+1} but non-adjacent to V_{c-1}, and if V_k is completely adjacent to V_{c-1} but non-adjacent to V_{a+1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j, \{v_{i+1}\}, V_k, V_d, \ldots, V_c, \ldots$$

If V_j is completely adjacent to V_{a+1} but non-adjacent to V_{c-1}, and if V_k is completely adjacent to V_{c-1} and V_{a+1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j, \{v_{i+1}\} \cup V_k, V_d, \ldots, V_c, \ldots$$

If V_j is completely adjacent to V_{c-1} and V_{a+1}, and if V_k is completely adjacent to V_{c-1} but non-adjacent to V_{a+1}, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j \cup \{v_{i+1}\}, V_k, V_d, \ldots, V_c, \ldots$$

Suppose that $d = b + 4$. Let V_j, V_k, and V_l, where $j = b + 1$, $k = b + 2$, and $l = b + 3$, be the three blocks between V_b and V_d. Then the only possible situation is the following: V_j is completely adjacent to V_{a+1} but non-adjacent to V_{c-1}, V_k is completely adjacent to both V_{c-1} and V_{a+1}, and V_l is completely adjacent to V_{c-1} but non-adjacent to V_{a+1}. In this case the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, V_j, V_k \cup \{v_{i+1}\}, V_l, V_d, \ldots, V_c, \ldots$$

Subcase 2.2. There exist S and S' with $\emptyset \neq S \subset V_a$ and $\emptyset \neq S' \subset V_c$ such that v_{i+1} is adjacent to every vertex in $S \cup S'$ but to no vertex in $(V_a - S) \cup (V_c - S')$.

Suppose that $c = a + 1$. Fact 3 implies that $a = 1$ and $c = p$, that is, V_a is the first block and V_c is the last block in the straight enumeration of the blocks of H_i. Since V_a and V_c are completely adjacent, $V_a \cup V_c$ must be a block in H_i, contradicting the hypothesis. Thus $c \geq a + 2$. By Fact 2, V_a is completely non-adjacent to V_c. Let $V_a V_b$ be the right-most wave at V_a and let $V_c V_d$ be the left-most wave at V_d. By Fact 3, $b < d$. Suppose that $d > b + 1$. Let V_j be a block between V_b and V_d, i.e., $b < j < d$. Then any choice of three vertices from S, S', V_j, respectively, together with v_{i+1} would induce a copy of the claw in G, a contradiction. Hence $d = b + 1$, i.e., there is no block between V_b and V_d. In this case, the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a - S, S, \ldots, V_b, \{v_{i+1}\}, V_d, \ldots, S', V_c - S', \ldots$$

Subcase 2.3. There exists $\emptyset \neq S \subset V_a$ such that v_{i+1} is adjacent to every vertex in S but to no vertex in $(V_a - S) \cup V_c$. (A similar discussion applies when there exists $\emptyset \neq S' \subset V_c$ such that v_{i+1} is adjacent to every vertex in S' but to no vertex in $(V_c - S) \cup V_a$.)

Suppose that V_a is completely adjacent to V_c. If there exists a block V_e which is completely adjacent to V_a but non-adjacent to V_c, then $e < a$ and any choice of three vertices from V_e, S, V_c, respectively, together with v_{i+1} induce a copy of the claw in G, a contradiction. So there exists a block V_f which is completely adjacent to V_c but not to V_a. Then $f > c$.

Let $V_c V_d$ be the left-most wave at V_c. Suppose that $a \neq 1$. Then $d \leq a$ and from the above discussion $V_a V_d$ also must be the left-most wave at V_a. Then $d \leq a - 1$. If $d \neq 1$, then any choice of vertices from $V_{d-1}, V_d, S, V_c, V_f$, respectively, together with v_{i+1} induce a subgraph of G which is not local-tournament-orientable, a contradiction. Assume that $d = 1$. Consider the block V_d and the block V_a. Then there must be a block V_g which is completely adjacent to V_a but not to V_d. We must have $g > c$ and hence any choice of three vertices x, y, z from V_d, V_a, V_g together with v_{i+1} induce a copy of the claw in G, a contradiction. Therefore $a = 1$ and the straight enumeration of the blocks of H_{i+1} is

$$\{v_{i+1}\}, S, V_a - S, V_c, \ldots$$

Suppose now that the block V_a is completely non-adjacent to the block V_c. Then $c > a + 1$. Let V_aV_b be the right-most wave at V_a and V_cV_d be the left-most wave at V_c. By Fact 3, $b < d$. We observe that for each block V_j with $b < j < d$, V_j is completely adjacent to V_{c-1}, as otherwise any choice of three vertices from S, V_j, V_{c-1}, respectively, together with v_{i+1} would induce a copy of the claw in G, a contradiction. Hence the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a - S, S, \ldots, V_b, \{v_{i+1}\}, \ldots, V_d, \ldots, V_c, \ldots.$$

Subcase 2.4. There exists $\emptyset \neq S \subset V_a$ such that v_{i+1} is adjacent to every vertex in $S \cup V_c$ but to no vertex in $V_a - S$. (A similar discussion applies when there exists S' such that $\emptyset \neq S' \subset V_c$ and v_{i+1} is adjacent to every vertex in $S' \cup V_a$ and to no vertex in $V_c - S'$.)

If V_c is not the last block, i.e., $c \neq p$, then it can be treated as Subcase 2.3. Suppose that V_c is the last block, namely, $c = p$. Let V_aV_b be the right-most wave at V_a and V_cV_d be the left-most wave at V_c.

Suppose that $b = c$. Then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a - S, S, \ldots, V_c, \{v_{i+1}\}$$

Suppose that $b < c$. If $d > b + 1$ and V_j is a block between V_b and B_d, then any choice of three vertices from S, V_j, V_c, respectively, together with v_{i+1} would induce a copy of the claw, a contradiction. Hence $d \leq b + 1$. If $d = b + 1$, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a - S, S, \ldots V_b, \{v_{i+1}\}, V_d, \ldots V_c$$

If $d < b + 1$, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a - S, S, \ldots, V_b, \{v_{i+1}\}, V_{b+1}, \ldots V_c$$

Subcase 2.5. Finally we consider the case when v_{i+1} is adjacent to every vertex in $V_a \cup V_c$. If V_a is not the first block and V_c is not the last block in the straight enumeration of the blocks of H_i, i.e., $a \neq 1$ and $c \neq p$, then it can be treated as in Subcase 2.1.

Suppose $a = 1$ and $c = p$. Note that V_a is completely non-adjacent to V_c. Let V_aV_b be the right-most wave at V_a and let V_cV_d be the left-most wave at V_c. If $d < b$, then, for any $x \in V_b$ and $y \in V_d$, $N[x] = N[y]$, a contradiction. Hence $d \geq b$. If $d = b$, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b \cup \{v_{i+1}\}, \ldots, V_c.$$

Suppose that $d > b$. If $d > b+1$ and V_j is a block between V_b and V_d, then any choice of three vertices from V_a, V_j, V_c, respectively, together with v_{i+1} induce a copy of the claw in G, a contradiction. Hence $d = b + 1$. Therefore the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_b, \{v_{i+1}\}, V_d, \ldots, V_c.$$

Suppose that V_a is not the first block and suppose that V_c is the last block in the straight enumeration of the blocks of H_i, namely, $a > 1$ and $c = p$ (a similar discussion applies if $a = 1$ and $c < p$). Let $V_{a-1}V_e$ be the right-most wave at V_{a-1}. Then $a \leq e \leq c$. If $e = c$, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_{a-1}, V_a, \ldots, V_c, \{v_{i+1}\}.$$

Suppose that $e < c$. Note that any block between V_e and V_c (if there is any) must be either completely adjacent to V_a or to V_c. Hence there are at most three blocks between V_e and V_c, that is, $c \leq e + 4$.

When $c = e+1$, there is no block between V_e and V_c. Then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, \{v_{i+1}\}, V_c.$$

For the case when $c = e + 2$, let V_j be the only block between V_e and V_c. We know that V_j has to be completely adjacent to at least one of V_a and V_c. If V_j is completely adjacent to both V_a and V_c, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j \cup \{v_{i+1}\}, V_c.$$

If V_j is completely adjacent to V_a but non-adjacent to V_c, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j, \{v_{i+1}\}, V_c.$$

If V_j is completely adjacent to V_c but non-adjacent to V_a, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, \{v_{i+1}\}, V_j, V_c.$$

Suppose that $c = e + 3$. Let V_j and V_k be the two blocks between V_e and V_c where $j = e + 1$ and $k = e + 2$. If V_j is completely adjacent to V_a but non-adjacent to V_c, and if V_k is completely adjacent to V_c but non-adjacent to V_a, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j, \{v_{i+1}\}, V_k, V_c.$$

If V_j is completely adjacent to both V_a and V_c, and if V_k is completely adjacent to V_c but non-adjacent to V_a, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j \cup \{v_{i+1}\}, V_k, V_c.$$

If V_j is completely adjacent to V_a but non-adjacent to V_c, and if V_k is completely adjacent to both V_a and V_c, then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j, V_k \cup \{v_{i+1}\}, V_c.$$

Suppose that $c = e + 4$. Let V_j, V_k, and V_l be the three blocks between V_e and V_c, where $j = e + 1$, $k = e + 2$, and $l = e + 3$. Then the only situation is the following: V_j is completely adjacent to V_a but non-adjacent to V_c, V_k is completely adjacent to both V_a and V_c, and V_l is completely adjacent to V_c but non-adjacent to V_a. Then the straight enumeration of the blocks of H_{i+1} is

$$\ldots, V_a, \ldots, V_e, V_j, V_k \cup \{v_{i+1}\}, V_l, V_c.$$

We now analyze the time cost of Step 2 of Algorithm 4.2.1. We show that it takes time $O(deg(v_{i+1}))$ to insert the vertex v_{i+1} into H_i. When H_i has only one block, it is clear. Suppose that H_i contains at least two blocks. (Note that in this case H_i must contain at least three blocks.) According to the above discussion, we need to find V_a and V_c where $a < c$ such that v_{i+1} is adjacent to all vertices in V_b for each b with $a < b < c$, and v_{i+1} is adjacent to no vertex in V_d for any d with $d < a$ or $d > c$. Let V_f be a block of H_i which contains a neighbour of v_{i+1}. We can find f from the knowledge of a vertex $v_j, j \leq i$ adjacent to v_{i+1} (cf. Step 1). If $f = 1$, then we let $a = f$. Choose a vertex x of V_{f-1}. If $f - 1 = 1$, then $a = 1$. Otherwise, if x is not a neighbour of v_{i+1}, then a must be either $f - 1$ or f. We can decide which of these two is the case, in time $O(deg(v_{i+1}))$, as follows: We test adjacency of v_{i+1} to individual elements of V_f, until we find the first element of V_f which is not adjacent to v_{i+1}. If such an element exists, then $a = f$; otherwise $a = f - 1$. If x is a neighbour of v_{i+1}, then we choose a vertex of V_{f-2}, and continue in this fashion, until we find vertices $y \in V_g, z \in V_{g+1}$ such that v_{i+1} is adjacent to z but not to y. Then $a = g$ or $a = g + 1$, and we decide as above. If v_{i+1} is adjacent to a vertex in each of the blocks $V_1, V_2, \ldots, V_f$, then $a = 1$. This procedure takes time $O(deg(v_{i+1}))$. Similarly in time $O(deg(v_{i+1}))$ we can find the block V_c.

At each stage, we keep track of enough information for the straight enumeration of the blocks of H_i, such as the left-most wave and the right-most wave at each block of H_i. After we have found the blocks V_a and V_c, we can obtain a straight-mixed-graph orientation H_{i+1} in time $O(deg(v_{i+1}))$, by considering the above cases. (The neighbours of v_{i+1} in any V_s can also be identified in time $O(deg(v_{i+1}))$.) Hence we can find a straight-mixed-graph orientation of G in time $O(m + n)$.

Theorem 4.2.2 *Algorithm 4.2.1 takes time $O(m + n)$ (in the worst case) to find a straight-mixed-graph orientation of G, or to correctly report that G is not a proper interval graph.* □

Suppose that H is a straight-mixed-graph orientation of G. If we orient each block of H transitively, then we obtain an acyclic local-tournament orientation of G. By applying the technique explained in the proof of Theorem 2.2.2, we obtain, in time

$O(m+n)$, an inclusion-free interval family associated with G. Therefore we have the following result.

Corollary 4.2.3 *The recognition and representation problem for proper interval graphs can be solved in time $O(m+n)$.* □

4.3 Proper Circular Arc Graphs

In this section, we give an $O(m+n)$ time algorithm for the recognition and representation of proper circular arc graphs. The idea of our algorithm is in fact to test if a graph is orientable as a local transitive tournament. We know that a graph is not a proper circular arc graph if it is not local-transitive-tournament orientable. If a graph is local-transitive-tournament-orientable, then a local-transitive-tournament orientation is obtained. By Corollary 2.2.7 a representation can be obtained in time $O(m+n)$ from a local-transitive-tournament orientation. Our algorithm involves an $O(m+n)$ algorithm for testing proper interval graphs and finding corresponding acyclic local-tournament orientations.

In view of Tucker's $O(n^2)$ time algorithm, we only need to deal with the case when the number of edges is small relative to n^2.

Algorithm 4.3.1 *Let G be a graph with n vertices and m edges.*

[Step 0.] Test if G is a proper interval graph. If it is, represent it by intervals (viewed as a special case of circular arcs).

[Step 1.] Choose a vertex x of minimum degree in G. Let A be the subgraph induced by $N[x]$ and let $B = G - A$. If B is a clique, solve the recognition and representation problems for G by Tucker's algorithm.

[Step 2.] Orient both graphs A and B as straight mixed graph. (This is unique (cf. Theorem 4.1.1).)

[Step 3.] Merge (cf. below) these orientations into a strong local-tournament orientation of the entire graph G.

[Step 4.] Modify the result into a local transitive tournament.

[Step 5.] Transform the local transitive tournament into a circular arc representation of G. □

Step 0 can be done in time $O(m+n)$ (see Section 4.2). Step 1 also takes time $O(m+n)$ because when B is a clique, the number of edges of G is $m \geq \frac{n^2-2n}{4}$ (recall that x is a vertex of minimum degree), and so Tucker's algorithm runs in time $O(m+n)$ in this case. The previous section covers Step 2. Step 5 can be carried out in time $O(m+n)$, as explained in Theorem 2.2.6.

Thus we only need to discuss Steps 3 and 4. Let G be a proper circular arc graph which is not a proper interval graph and for which B is not a clique.

Proposition 4.3.2 *Both A and B are connected proper interval graphs.*

Proof: Since B is not a clique, any proper circular arc representation of G contains three disjoint circular arcs – one corresponding to x and two corresponding to two non-adjacent vertices of B. Let X be a point on the circular arc corresponding to x. The other two circular arcs divide the circle into two segments. Choose a point Y on that segment which does not contain X. Then no circular arc in B contains the point X and no circular arc in A contains Y. Thus both A and B are proper interval graphs. Since G is not a proper interval graph, A and B are connected. □

Proposition 4.3.3 *The graph G is uniquely orientable as a mixed local tournament.*

Proof: The above three disjoint circular arcs correspond to a triangle in the complement of G. Hence the orientation is essentially unique by Proposition 2.3.10. □

From Theorem 4.1.1, we know that A and B are uniquely orientable as straight mixed graphs. We consider the following two cases.

Case 1. Suppose that A is not a clique. Then, in the mixed-graph orientation of A, let L be the set of vertices in A which are not in the outset of x, and let R be the set of vertices of A which are not in the inset of x. Let C be the graph induced by B and L, and let D be the graph induced by B and R. Since G is not an interval graph, both C and D are connected. It is also easy to see that both C and D are proper interval graphs: it is enough to choose points Z and W as the leftmost and rightmost endpoints of the intervals representing x and all vertices with the same closed neighbourhood as x in A. No circular arc of C contains the point W and no circular arc of D contains Z. Now all four graphs A, B, C, D can be uniquely oriented as straight mixed graphs. Consider $\vec{G}$, an orientation of G, as a mixed local tournament. Of the two possible orientations of A (and similarly for B, C, and D), one must agree with $\vec{G}$ in the sense that any edge oriented in A is oriented in $\vec{G}$ in the same direction. Therefore, if we choose one of the two orientations of A and one of C, either the edges oriented in both all agree or all disagree in direction. Thus we may choose orientations $\vec{A}, \vec{B}, \vec{C}, \vec{D}$ such that any edges oriented in two (or more) agree in their direction.

Theorem 4.3.4 *The oriented edges of the mixed local tournament $\vec{G}$ are precisely the union of the oriented edges of $\vec{A}, \vec{B}, \vec{C}, \vec{D}$.*

Proof: If an edge uv is oriented in $\vec{G}$ then u and v are not equivalent, i.e., have distinct closed neighbourhoods in the underlying graph of $\vec{G}$. Suppose both u and v belong to A, and have the same neighbours in A (otherwise uv is oriented in $\vec{A}$). Then u and v are both in R or both in L. Suppose they are both in L and thus both in C. Since they are not equivalent in G and are equivalent in A, they must be not equivalent in C. The other cases (one in A one in B or both in B) are similar. Therefore any edge oriented in $\vec{G}$ is oriented in at least one of $\vec{A}$, $\vec{B}$, $\vec{C}$, $\vec{D}$.

Let uv be oriented in $\vec{A}, \vec{B}, \vec{C}$ or $\vec{D}$. Then the neighbourhoods of u and v are distinct in that graph, and hence certainly also distinct in G. Therefore uv is also oriented in $\vec{G}$. Since we observed above that there are no conflicts in the orientations, the proof is complete. □

Thus Step 3 is done by orienting C and D and then combining the orientations of A, B, C, D as above. It is clear that each of these steps can be performed in time $O(m + n)$.

Step 4 is then accomplished by defining an arbitrary transitive tournament on the vertices of each equivalence class of the mixed local tournament $\vec{G}$.

Case 2. Suppose now that A is a clique. Suppose that $V_1, V_2, \ldots, V_k$ is the straight enumeration of the blocks of B. Since B is not a clique, we know that $k \geq 3$. Let L be the set of vertices in A which are adjacent to at least one vertex in V_1, and let R be the set of vertices in A which are adjacent to at least one vertex in V_k (note that these vertex sets can be found in time $O(m + n)$). Let C be the graph induced by B and L, and D be the graph induced by B and R. We follow the procedures as above by considering A, B, C, and D. Again A, B, C, and D are proper interval graphs and hence they can be oriented uniquely as a straight mixed graph. If we choose orientations $\vec{A}, \vec{B}, \vec{C}, \vec{D}$ such that any edges oriented in two (or more) agree in their direction, then we can apply a proof similar to the proof of Theorem 4.3.4 to show that the union of the oriented edges in $\vec{A}, \vec{B}, \vec{C}, \vec{D}$, and $L \rightarrow R$ give a mixed-local-tournament orientation $\vec{G}$ of G. Therefore a local-transitive-tournament orientation of G can be obtained from $\vec{G}$. This completes the proof of correctness of Algorithm 4.3.1.

There exist efficient algorithms for solving many basic optimization problems for proper circular arc graphs which assume that a proper circular arc representation is given. For instance, Hsu and Tsai [46] have an $O(n)$ algorithm to find a maximum independent set and to find a minimum clique covering in a proper circular arc graph. (In fact, the algorithm applies in a general circular arc graph.) In view of our $O(m+n)$ representation algorithm, we may now conclude that the maximum independent set problem and the minimum clique covering problem for proper circular arc graphs are solvable in time $O(m + n)$.

Chapter 5

Maximum Cliques and c-Colourings

In this chapter, we will give two algorithms: one is an $O(m+n)$ time algorithm to find a maximum clique of a proper circular arc graph, and the other is an $O(m+n)$ algorithm to determine c-colourability of a proper circular arc graph. Again these algorithms do not require an arc representation, but can be implemented in time $O(n \log n)$ if such a representation is given.

Proper circular arc graphs have applications in traffic control [74] cyclic scheduling and compiler design, [79]. The problem of finding the maximum clique in general circ lar arc graphs has been previously solved by Apostolico and Hambrusch [3], by an algorithm which has a time bound of $O(n^2 \log\log n)$. However this algorithm requires that the representation by a circular arc family be given. For the special case of proper circular arc graphs, we shall give here an $O(m+n)$ algorithm. If the representation is known, our algorithm can be implemented to run in time $O(n \log n)$.

The problem of c-colouring proper circular arc graphs arose in the cyclic scheduling and register allocation applications. It was first studied by Orlin, Bonuccelli and Bovet [61]. Their approach consisted of reducing the problem to a shortest path calculation, and resulted in an algorithm with a time bound of $O(n^2)$. Subsequently, other authors improved the algorithm by improving on the shortest path method, culminating in the algorithm of Shih and Hsu [70], which has a time bound of $O(n^{\frac{3}{2}})$. However all these algorithms require the representation by a circular arc family to be given. By applying our maximum clique algorithm we are also able to give an $O(m + n)$ algorithm for this problem. If the representation is known, our algorithm can be implemented to run also in time $O(n \log n)$.

We first remark that we may assume that G is connected and that it has no vertices of degree $n - 1$. Indeed, both the colouring and the maximum clique problems can be solved for each component separately. Furthermore, any maximum clique must contain all vertices of degree $n - 1$, and any colouring must assign each vertex of degree $n - 1$ a colour not used by any other vertex. Thus it is sufficient to solve both problems for the graph obtained by removing all vertices of degree $n - 1$. Therefore we shall assume from now on that G is a connected graph with $\Delta(G) \leq n - 2$.

The new element in our approach is Theorem 2.1.15 which allows us to search for maximum cliques and minimum colourings in a more efficient way. From Section 4.3, we can obtain in time $O(m + n)$ a local transitive tournament orientation of any proper circular arc graph and hence in time $O(m + n)$ a round enumeration of the corresponding local transitive tournament.

Suppose that D is a local-transitive-tournament orientation of G. Then D is a round oriented graph. Let $v_1, v_2, \ldots, v_n$ be a round enumeration of D, such that for each i there exist non-negative integers l_i and r_i with $v_i \rightarrow v_j$ if and only if $i + 1 \leq j \leq i + r_i$ and $v_k \rightarrow v_i$ if and only if $i - l_i \leq k \leq i - 1$ (with the additions and subtractions modulo n). We define $R(u) = w$ just if $u = v_i$ and $w = v_{i+r_i}$, and $L(u) = w$ just if $u = v_i$ and $w = v_{i-l_i}$.

Note that the assumption that $\Delta(D) \leq n - 2$ implies that for every vertex u there is at least one non-neighbour of u between $R(u)$ and $L(u)$. Thus for every u moving clockwise we first encounter all out-neighbours of u (the last being $R(u)$), then all non-neighbours of u (of which there is at least one) and finally, just before returning to u, all in-neighbours of u (the first one being $L(u)$). In particular, for each vertex $u = v_i$, the set $\{u = v_i, v_{i+1}, v_{i+2}, \ldots, v_{i+r_i} = R(u)\}$ induces a clique (in fact a transitive tournament of D).

In the sequel, we show how searching for maximum cliques and for minimum colourings in G can be made more efficient if we perform it on a round enumeration of a local transitive orientation D of G.

Let $v_1, v_2, \ldots, v_n$ be a round enumeration of a local transitive tournament D. Let $a = v_i$ and $b = v_j$. The *interval* $[a, b]$ is the set of vertices $\{v_i, v_{i+1}, v_{i+2}, \ldots, v_{j-1}, v_j\}$, with the subscripts calculated modulo n. (Thus if we draw $v_1, v_2, \ldots, v_n$ clockwise around the circle, the interval $[a, b]$ extends from a to b clockwise.) The intervals $(a, b), (a, b]$ and $[a, b)$ are defined analogously.

We observe that if $a \rightarrow b$ then $x \rightarrow y$ for all x, y with $[x, y] \subseteq [a, b]$, and so $[a, b]$ is a complete graph (in fact, a transitive tournament).

A *clique* of a graph (or an oriented graph) is just a complete subgraph. A clique of maximum size is called a *maximum clique*.

5.1 The Maximum Clique Algorithm

Assume that D is a local transitive tournament with a round enumeration $v_1, v_2, \ldots, v_n$. We shall restrict our search for a maximum clique in D to a special class of cliques defined as follows: Let m be an odd integer $m \geq 3$, and let $a_1, b_1, a_2, b_2, \ldots, a_m, b_m$ be distinct vertices of D listed in clockwise circular order, such that for each $i = 1, 2, \ldots, m$ we have

$$R(a_i) = b_{i+m'} \quad \textit{and} \quad |[a_i, b_i]| > |(b_{i+m'}, a_{i+m'+1})|,$$

where $m' = \frac{m-1}{2}$ and the subscript additions are modulo m. Then we say that $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ is an *m-overlap clique generated by* $\{a_1, a_2, \ldots, a_m\}$. We also refer to the vertices $a_1, a_2, \ldots, a_m$ as the *generators* of C.

It is possible to specify an m-overlap clique by its generators. The generators must be distinct vertices $a_1, a_2, \ldots, a_m$ listed in clockwise circular order; we introduce $b_i = R(a_{i-m'})$ and verify that each $b_i \in (a_i, a_{i+1})$. If we also have $|[a_i, b_i]| > |(b_{i+m'}, a_{i+m'+1})|$, then $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ is an m-overlap clique generated by the given $a_1, a_2, \ldots, a_m$. It follows that between any two successive generators a_i, a_{i+1} of an m-overlap clique, there must be some $R(a_j)$ for a generator a_j. Another useful property to observe is that for any two generators a_i, a_j of an m-overlap clique we have $[a_i, R(a_i)] \cap [a_j, R(a_j)] \neq \emptyset$. Finally, we also note that in an m-overlap clique $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ we must have each $(b_i, a_{i+1}) \neq \emptyset$ because $a_{i+m'+1} = a_{i-m'}$ dominates b_i and is dominated by a_{i+1} (as $a_{i+1} \rightarrow b_{i+m'+1}$), so that in order to have the degree of $a_{i-m'}$ smaller than $n-1$, we need $(b_i, a_{i+1}) \neq \emptyset$.

For convenience we also define 1- and (-1)-overlap cliques: A 1-overlap clique is any interval $[a, b]$ with $b = R(a)$. Thus the interval $[a, R(a)]$ is the 1-overlap clique generated by a. (This coincides with the definition of an "overlap clique" in [61]). A (-1)-overlap clique is just the empty set $\emptyset$.

Lemma 5.1.1 *In the digraph D, we have:*

- *Any m-overlap clique* is *a clique;*
- *There exists a maximum clique which is an m-overlap clique for some odd m.*

Proof: The first statement clearly holds for 1- and (-1)- overlap cliques. Thus let $[a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ be an m-overlap clique of D with $m \geq 3$. Let $u \in [a_i, b_i]$. Since a_i dominates $b_{i+m'}$, the vertex u dominates all vertices of $(u, b_i] \cup [a_{i+1}, b_{i+1}] \cup \ldots [a_{i+m'}, b_{i+m'}]$. Since $b_{i+2m'+1} = b_i$ is dominated by $a_{i+m'+1}$, the vertex u is dominated by all vertices of $[a_{i+m'+1}, b_{i+m'+1}] \cup \ldots \cup [a_i, u)$. Therefore $[a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ is a clique of D. (Recall that $m' = \frac{m-1}{2}$, so that $2m' + 1 = m$.)

To prove the second statement, consider a set of vertices C which induces a maximum clique of G. If $C \neq \emptyset$, then there exists an integer m such that C may be written as $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ where $a_1, b_1, a_2, b_2, \ldots, a_m, b_m$ appear in clockwise circular order in the round enumeration. Let C and m be chosen so that m is as small as possible among all maximum cliques of G.

Suppose that $m = 1$, i.e., that $C = [a_1, b_1]$. Consider the adjacent vertices a_1, b_1. If $b_1 \rightarrow a_1$, then $[b_1, a_1]$ is also a clique, contradicting $\Delta(D) \leq n-2$. Thus $a_1 \rightarrow b_1$, and hence $R(a_1) \in [b_1, a_1)$. Since $[a_1, b_1]$ is a maximum clique, $R(a_1) = b_1$ and C is a 1-overlap clique. Thus suppose for the rest of this proof that $m > 1$.

Claim 1. If $u \notin [a_i, b_i]$ is adjacent to all vertices of $[a_i, b_i]$, then $u \to [a_i, b_i]$ or $[a_i, b_i] \to u$.

Suppose there are two vertices $x, y \in [a_i, b_i]$ such that $u \to x$ and $y \to u$. This means that $u \to (u, x]$ and $[y, u) \to u$. Together with the assumption that u is adjacent to each vertex of $[a_i, b_i]$, we contradict the fact that $\Delta(D) \leq n - 2$.

Claim 2. If $i \neq j$ then either $[a_i, b_i] \to [a_j, b_j]$ or $[a_j, b_j] \to [a_i, b_i]$.

If $a_i \to b_j$ then $x \to y$ for each $x \in [a_i, b_i]$ and $y \in [a_j, b_j]$. On the other hand, if $b_j \to a_i$ then two applications of Claim 1 yield $b_j \to b_i$ and $a_j \to b_i$. Thus in this case $x \to y$ for each $x \in [a_j, b_j]$ and $y \in [a_i, b_i]$.

Claim 3. If $[a_i, b_i] \to [a_j, b_j]$ then $[a_{i+1}, b_{i+1}] \to [a_{j+1}, b_{j+1}]$.

Suppose that $[a_i, b_i] \to [a_j, b_j]$ and $[a_{j+1}, b_{j+1}] \to [a_{i+1}, b_{i+1}]$. Let $u \in (b_i, a_{i+1})$. (It was noted above that $(b_i, a_{i+1}) \neq \emptyset$.) Then u is adjacent to all vertices of $[a_{i+1}, b_{i+1}] \cup \ldots \cup [a_j, b_j]$ because $a_i \to b_j$, and to all vertices of $[a_{j+1}, b_{j+1}] \cup \ldots \cup [a_i, b_i]$ because $a_{j+1} \to b_{i+1}$. This contradicts the maximality of our clique.

Claim 4. m is odd.

If m is even, then $[a_i, b_i] \to [a_{i+m/2}, b_{i+m/2}]$ implies $[a_{i+m/2}, b_{i+m/2}] \to [a_i, b_i]$ by Claim 3, contrary to Claim 2.

Claim 5. $R(a_i) = b_{i+m'}$ for each $i = 1, 2, \ldots, m$.

Since a_i and $b_{i+m'}$ are in C, they are adjacent. If some $b_{i+m'} \to a_i$, then Claim 2 implies that $[a_{i+m'}, b_{i+m'}] \to [a_i, b_i]$ and Claim 3 implies that $[a_i, b_i] \to [a_{i+m'+1}, b_{i+m'+1}]$. However, this is impossible as $a_i \to b_{i+m'+1}$ implies $a_i \to b_{i+m'}$. Hence $a_i \to b_{i+m'}$ for each $i = 1, 2, \ldots, m$. In particular, $a_{i+m'+1} \to b_{i+2m'+1} = b_i$. So $R(a_i) \in [b_{i+m'}, a_{i+m'+1})$. On the other hand, if $R(a_i) \neq b_{i+m'}$, then $R(a_i)$ is adjacent to every vertex of $[a_i, b_i] \cup [a_{i+1}, b_{i+1}] \cup \ldots \cup [a_{i+m'}, b_{i+m'}]$, and because $a_{i+m'} \to b_{i+2m'} = b_{i-1}$, $R(a_i)$ is also adjacent to every vertex of $[a_{i+m'+1}, b_{i+m'+1}] \cup \ldots \cup [a_{i-1}, b_{i-1}]$. Thus $R(a_i)$ is adjacent to every vertex of C, contradicting its maximality. Therefore $R(a_i) = b_{i+m'}$.

Claim 6. $|[a_i, b_i]| > |(b_{i+m'}, a_{i+m'+1})|$ for each $i = 1, 2, \ldots, m$.

If $|[a_i, b_i]| \leq |(b_{i+m'}, a_{i+m'+1})|$ for some i, then let $C' = [a_1, b_1] \cup \ldots \cup [a_{i-2}, b_{i-2}] \cup [a_{i-1}, b_i] \cup [a_{i+1}, b_{i+1}] \cup \ldots \cup [a_{i+m'-1}, b_{i+m'-1}] \cup [a_{i+m'}, b_{i+m'+1}] \cup [a_{i+m'+2}, b_{i+m'+2}] \cup \ldots [a_m, b_m]$. In effect, C' is obtained from C by replacing $[a_i, b_i]$ with $(b_{i+m'}, a_{i+m'+1})$. We see easily that C' is also a clique. We only need to verify that each vertex

$u \in (b_{i+m'}, a_{i+m'+1})$ is adjacent to all other vertices of C'. Since $a_{i+1} \rightarrow b_{i+m'+1}$ and hence $a_{i+1} \rightarrow u$, we conclude u is adjacent to $[ai+1, b_{i+1}] \cup \ldots \cup [a_{i+m'-1}, b_{i+m'-1}] \cup [a_{i+m'}, u)$; since $a_{i+m'} \rightarrow b_{i-1}$ and hence $u \rightarrow b_{i-1}$, we also conclude that u is adjacent to $(u, b_{i+m'+1}] \cup \ldots \cup [a_{i-1}, b_{i-1}]$. Thus C' is a clique with fewer intervals than C and with $|C'| \geq |C|$, contradicting the choice of C. □

Note that the converse of the second statement of Lemma 5.1.1 is not true, namely, an m-overlap clique is not necessarily a maximum clique. In fact, there may be m-overlap cliques of different sizes. We call an m-overlap clique of maximum size a *largest m-overlap clique.*

Lemma 5.1.2 *Let $m \geq 3$ be an odd integer. Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ be an m-overlap clique of D and suppose that $x \in (b_{i-1}, b_i]$ for some $i = 1, 2, \ldots, m$.*

If $|[x, R(x)]| \geq |[a_i, R(a_i)]|$, then the vertices $a_1, \ldots a_{i-1}, x, a_{i+1}, \ldots, a_m$ generate an m-overlap clique C', with $|C'| \geq |C|$. Moreover, $|C'| = |C|$ if and only if $|[x, R(x)]| = |[a_i, R(a_i)]|$.

Proof: To prove that C' is an m-overlap clique we need to show

1. $x \in (b_{i-1}, b_i)$ (that is, $x \neq b_i$),
2. $R(x) \in (a_{i+m'}, a_{i+m'+1})$,
3. $|[x, b_i]| > |(R(x), a_{i+m'+1})|$, and
4. $|[a_{i+m'}, R(x)]| > |(b_{i-1}, x)|$.

By the assumption, $x \in (b_{i-1}, b_i]$. If $x = b_i$, then $x \rightarrow b_{i+m'}$ and $a_{i+m'+1} \rightarrow x$. Thus $R(x) \in [b_{i+m'}, a_{i+m'+1})$. Since $|[x, R(x)]| \geq |[a_i, R(a_i)]|$, then $|[b_{i+m'}, R(x)]| \geq |[a_i, x]|$. Note that $\Delta(D) \leq n-2$ implies that $(R(x), a_{i+m'+1}) \neq \emptyset$. Then $|(b_{i+m'}, a_{i+m'+1})| \geq |[b_{i+m'}, R(x)]| \geq |[a_i, x]|$ (recall that $x = b_i$), contradicting the fact that C is an m-overlap clique. Hence $x \neq b_i$ and 1 holds.

Now we consider the vertex $R(x)$. Since $a_{i-1} \rightarrow b_{i+m'-1}$, we have $x \rightarrow b_{i+m'-1}$. Since $a_{i+m'+1} \rightarrow b_i$, we also have $a_{i+m'+1} \rightarrow x$. Thus $R(x) \in [b_{i+m'-1}, a_{i+m'+1})$. We claim that

$R(x) \notin [b_{i+m'-1}, a_{i+m'}]$. Suppose, to the contrary, that $R(x) \in [b_{i+m'-1}, a_{i+m'}]$. Then $x \in (b_{i-1}, a_i)$ because $a_i \rightarrow b_{i+m'}$. Since $|[x, R(x)]| \geq |[a_i, R(a_i)]|$, we have $|[x, a_i]| \geq |[R(x), R(a_i)]|$. We consider two cases: first if $R(x) \neq a_{i+m'}$, then $|(b_{i-1}, a_i)| \geq |[a_{i+m'}, b_{i+m'}]|$, contradicting the fact that C is an m-overlap clique; secondly if $R(x) = a_{i+m'}$, then, noticing that the assumption $\Delta(D) \leq n-2$ implies that $(b_{i-1}, x) \neq \emptyset$, we again have $|(b_{i-1}, a_i)| \geq |[a_{i+m'}, b_{i+m'}]|$, contradicting the fact that C is an m-overlap clique. This proves 2.

Finally we prove 3 and 4 together.

If $x \in (b_{i-1}, a_i]$, then $R(x) \notin (b_{i+m'}, a_{i+m'+1})$ as otherwise we would have $a_i \rightarrow R(x)$ contradicting the fact that $R(a_i) = b_{i+m'}$. Thus $R(x) \in (a_{i+m'}, b_{i+m'}]$. Then $|[x, a_i)| \geq |(R(x), b_{i+m'}]|$ because $|[x, R(x)]| \geq |[a_i, R(a_i)]|$, and hence

$$|[x, b_i]| = |[x, a_i)| + |[a_i, b_i]| > |(R(x), b_{i+m'}]| + |(b_{i+m'}, a_{i+m'+1})| = |(R(x), a_{i+m'+1})|,$$

with a similar proof for $|(b_{i-1}, x)| < |[a_{i+m'}, R(x)]|$. On the other hand, if $x \in (a_i, b_i)$ then $R(x) \in (b_{i+m'}, a_{i+m'+1})$ and $|(b_{i+m'}, R(x)]| \geq |[a_i, x)|$ because $|[x, R(x)]| \geq |[a_i, R(a_i)]|$. Thus

$$|[x, b_i]| = |[a_i, b_i]| - |[a_i, x)| > |(b_{i+m'}, a_{i+m'+1})| - |(b_{i+m'}, R(x)]| = |(R(x), a_{i+m'+1})|,$$

with a similar proof for $|(b_{i-1}, x)| < |[a_{i+m'}, R(x)]|$.

It is now easy to conclude that $|C'| \geq |C|$ because $|[x, R(x)]| \geq |[a_i, R(a_i)]|$ means $|[x, a_i]| \geq |[R(x), R(a_i)]|$ (or $|[a_i, x]| \leq |[R(a_i), R(x)]|$); similarly we can conclude that $|C'| = |C|$ if and only if $|[x, R(x)]| = |[a_i, R(a_i)]|$. □

Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ be an m-overlap clique. We say that C is *localized* if for every $i = 1, 2, \ldots, m$ and each $x \in (b_{i-1}, b_i]$ we have $|[x, R(x)]| \leq [a_i, R(a_i)]|$. Note that the (-1)-overlap clique $\emptyset$ is localized, as is each largest 1-overlap clique.

We derive the next result from Lemma 5.1.2.

Corollary 5.1.3 *Let $m \geq -1$ be an odd integer. Every largest m-overlap clique is localized.* □

Corollary 5.1.4 *Let $m \geq 3$ be an odd integer. If $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ is a largest m-overlap clique, then for any $x \in (b_{i-1}, b_i]$, with $|[x, R(x)]| = |[a_i, R(a_i)]|$, the vertices $a_1, \ldots a_{i-1}, x, a_{i+1}, \ldots, a_m$ generate an m-overlap clique C', which also is a largest m-overlap clique.* □

Suppose k is the smallest integer such that there exists a maximum clique C of D which is a k-overlap clique. Then any largest k-overlap clique is a maximum clique. We shall assume k is fixed from now on, and denote $k' = \frac{k-1}{2}$.

Let K be a localized m-overlap clique of D, for some $m \leq k$. We say that K is *admissible* if there exists a largest k-overlap clique (hence a maximum clique) C of D such that each generator of K is also a generator of C. We also say that C is a *certificate* of admissibility of K. Note that an admissible clique is by definition localized.

Our strategy in searching for a maximum clique of D is to find an admissible 1-overlap clique, then to modify it to an admissible 3-overlap clique, then to an admissible 5-overlap clique, and so on, terminating with an admissible k-overlap clique which is also a maximum clique. The following lemma explains how to obtain an admissible 1-overlap clique.

Lemma 5.1.5 *Each largest 1-overlap clique is admissible.*

Proof: Suppose $[x, R(x)]$ is a largest 1-overlap clique. By Corollary 5.1.3, $[x, R(x)]$ is localized. Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ be any largest k-overlap clique (and thus a maximum clique of) D. Then $x \in (b_{i-1}, b_i]$ for some i, and hence $|[x, R(x)]| = |[a_i, R(a_i)]|$ because C is localized and $[x, R(x)]$ is a largest 1-overlap clique. Therefore by Corollary 5.1.4 the k-overlap clique C' generated by $a_1, \ldots, a_{i-1}, x, a_{i+1}, \ldots, a_k$ is a certificate of $[x, R(x)]$. □

Let $m \geq 1$ be an odd integer. Suppose $K = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_m, b_m]$ is a localized m-overlap clique of D, and suppose that there exist vertices c, d and subscript $i = 1, 2, \ldots, m$ such that

- $c \in (a_i, b_i), d \in (b_{i+m'}, a_{i+m'+1})$
- $R(c) \in (d, a_{i+m'+1}), R(d) \in (a_i, c)$
- $|(R(d), c)| < |[d, R(c)]|$.

Let $K^* = [a_1, b_1] \cup \ldots \cup [a_i, R(d)] \cup [c, b_i] \cup [a_{i+1}, b_{i+1}] \cup \ldots \cup [a_{i+m'}, b_{i+m'}] \cup [d, R(c)] \cup [a_{i+m'+1}, b_{i+m'+1}] \cup \ldots \cup [a_m, b_m]$. We say that K^* is a *modification of* K *obtained by replacing* $(R(d), c)$ *with* $[d, R(c)]$. If $m = -1$, we say that any 1-overlap clique K^* is a modification of the (-1)-overlap clique $\emptyset$. Let $|H|$ denote the number of vertices of H.

Lemma 5.1.6 *Let $m \geq -1$ be an odd integer, and let K be a localized m-overlap clique. Each modification K^* of K is an $(m+2)$-overlap clique, and $|K^*| > |K|$.*

Proof: This is clear for $m = -1$. For $m \geq 1$, it suffices to show that $|[c, b_i]| > |(R(c), a_{i+m'+1})|$ and $|[a_i, R(d)]| > |(b_{i+m'}, d)|$. If $|[c, b_i]| \leq |(R(c), a_{i+m'+1})|$, then $|(R(d), b_i]| < |[d, a_{i+m'+1})|$. Hence $|[d, R(d)]| > |[a_{i+m'+1}, b_i]|$. Since $R(a_{i+m'+1}) = b_i$ this contradicts the hypothesis that K is localized. A similar argument applies to show $|[a_i, R(d)]| > |(b_{i+m'}, d)|$. □

Note that the modification K^* of K has $|K^*| > |K|$. It follows that, in particular, a largest k-overlap clique (which is necessarily a maximum clique) admits no modification. There may of course be several possible modifications for a given K. A *localized modification* of K is any modification K^* of K which is itself localized. In particular, a localized modification of $\emptyset$ is any largest 1-overlap clique.

The importance of localized modifications to our algorithm is underscored by the following crucial result.

Theorem 5.1.7 *Let $m \geq -1$ be an odd integer. If K is an admissible m-overlap clique, then any localized modification K^* of K is an admissible $(m+2)$-overlap clique.*

Proof: Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ be a certificate for K. Assume that K^* is a modification of K obtained from K by replacing $(R(d), c)$ with $[d, R(c)]$. Then

K^* is an $(m+2)$-overlap clique by Lemma 5.1.6. We may assume that $c \in (a_p, b_r)$ where $[a_p, b_r]$ is one of the m intervals defining K. (Note that the generators of K are included among the generators of C.) It follows from the definition of K^* that $d \in (b_{p+k'}, a_{r+k'+1})$, where $(b_{p+k'}, a_{r+k'+1})$ is one of the open intervals separating two of the defining intervals of K. Similarly, $R(c) \in (d, a_{r+k'+1})$ and $R(d) \in (a_p, c)$.

We show that K^* is admissible. Without loss of generality, let $c \in (b_{i-1}, b_i]$ and $d \in (b_{j-1}, b_j]$. We shall show that we can alter C by removing both generators c, d and inserting the generators a_i, a_j, obtaining a certificate C^* of K^*. To do this, by Corollary 5.1.4, we need to prove $|[c, R(c)]| = |[a_i, R(a_i)]|$ and $|[d, R(d)]| = |[a_j, R(a_j)]|$. Since C is localized according to Corollary 5.1.3, $|[c, R(c)]| \leq |[a_i, R(a_i)]|$ and $|[d, R(d)]| \leq |[a_j, R(a_j)]|$. Thus we only need to show that $|[c, R(c)]| \geq |[a_i, R(a_i)]|$ and $|[d, R(d)]| \geq |[a_j, R(a_j)]|$. So it suffices to show that $a_i \in (R(d), b_r]$ and $a_j \in (b_{p+k'}, R(c)]$ as K^* is localized. Since the two cases are similar, we only show that $a_i \in (R(d), b_r]$.

If $a_i \notin (R(d), b_r]$, then $a_i \in (b_{i-1}, R(d)]$. Thus $R(d) \in [a_i, b_i]$ because $c \in (R(d), b_i]$. Hence $R(c) \in (d, a_{i+k'+1})$. Since $a_i \rightarrow b_{i+k'}$ and $a_{i+k'+1} \rightarrow b_i$, we have $d \in (b_{i+k'}, a_{i+k'+1}]$. If $c = b_i$, noting that $(R(c), a_{i+k'+1}) \neq \emptyset$, then we have $|(R(c), a_{i+k'+1})| \geq |[c, b_i]| = 1$. Combining this with the fact that $|[d, R(c)]| > |(R(d), c)|$, we have $|[d, a_{i+k'+1})| > |(R(d), b_i]|$. Thus $|[d, R(d)]| > |[a_{i+k'+1}, R(a_{i+k'+1})]|$ (recall that $R(a_{i+k'+1}) = b_i$), contradicting the fact that C is localized. If $R(d) = a_i$, in a similar way, we will have $|[c, R(c)]| > |[a_i, R(a_i)]|$, again contradicting the fact that C is localized. Hence $c \neq b_i$ and $R(d) \neq a_i$ and so C has a modification which can be obtained by replacing $(R(d), c)$ with $[d, R(c)]$, contradicting the fact that C is a maximum clique. □

Let $r(x) = |[x, R(x)]|$. We define for each pair of vertices x, y, the quantities $M(x,y) = max\{r(z) |\ z \in (x,y)\}$, $M[x,y] = max\{r(z) |\ z \in [x,y]\}$, and $M[x,y) = max\{r(z) |\ z \in [x,y)\}$.

Theorem 5.1.8 *Admissible cliques have the following properties:*

- *Any admissible m-overlap clique with $m < k$ admits a localized modification;*
- *Any admissible k-overlap clique is a maximum clique.*

Proof: If K is an admissible m-overlap clique, then $m \leq k$. Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ be a certificate of K. It is easy to see that if $m = k$, then $K = C$ and K admits no modification or localized modification.

Suppose that $m < k$. Then there exist p and r with $p \neq r$ such that $[a_p, b_r]$ is one of m intervals defining K (note that the generators of K are also generators of C). Let $f \in (a_p, b_r)$ be the first vertex in the order from a_p to b_r such that, for any g with $g \in (b_{p+k'}, R(f))$ and $r(g) = M(b_{p+k'}, R(f))$, we have the following properties:

- $R(f) \in (b_{p+k'}, a_{r+k'+1})$,
- $R(g) \in (a_p, f)$, and
- $|[g, R(f)]| > |(R(g), f)|$.

We note that such a vertex f exists because $a_{p+1} \in (a_p, b_r)$ satisfies these three properties. Therefore we also have $f \in (a_p, a_{p+1}]$. If $f \in (a_p, b_p)$, then $R(f) \in (b_{p+k'}, a_{p+k'+1})$ and hence C can be modified by replacing $(R(g), f)$ with $[g, R(f)]$, contrary to the maximality of C. Suppose now that $f = b_p$. Then again $R(f) \in (b_{p+k'}, a_{p+k'+1})$, and by the hypothesis that $\Delta(D) \leq n-2$, $|(R(f), a_{p+k'+1})| \geq 1$. Hence $|[g, a_{p+k'+1})| > |(R(g), b_p]|$, which implies that $|[g, R(g))| > |[a_{p+k'+1}, R(a_{p+k'+1}))|$ (since $R(a_{p+k'+1}) = b_p$), contradicting the fact that C is localized. Therefore $f \in (b_p, a_{p+1}]$. Now let c, d be vertices such that $r(c) = M[f, b_r)$ and $r(d) = M(b_{p+k'}, R(c))$. We will show that a localized modification of K_m can be obtained from K_m by replacing $(R(d), c)$ with $[d, R(c)]$.

Since $c \in [f, b_r)$, we have $c \in (b_{i-1}, b_i]$ where $i \in \{p+1, p+2, \ldots, r\}$. Then $r(a_i) = r(c)$ because $r(c) = M[f, b_r)$ and $r(a_i) = M(b_{i-1}, b_i] = M[f, b_r)$ (as $a_i \in [f, b_r)$ and C is localized). Hence Corollary 5.1.4 implies that $c \in (a_{i-1}, b_i)$ and $R(c) \in (a_{i+k'}, a_{i+k'+1})$. Suppose that $d \in (b_{j-1}, b_j]$ for some j. Then $j \in \{p + k' + 1, p + k' + 2, \ldots, i + k' + 1\}$. First we claim that $j \neq i + k' + 1$. Assume to the contrary that $j = i + k' + 1$. Then $d \in (b_{i+k'}, b_{i+k'+1}]$ and hence $d \in (b_{i+k'}, R(c))$. This implies that $R(c) \in (b_{i+k'}, a_{i+k'+1})$ and hence $c \in (a_i, b_i)$. Since $d \rightarrow b_{i-1}$ (as $a_{i+k'} \rightarrow b_{i-1}$) and $c \rightarrow d$ (as $a_{i+k'} \rightarrow b_{i-1}$), we see that $R(d) \in [b_{i-1}, c)$. If $R(d) \in [b_{i-1}, a_i)$, then $r(a_{i+k'}) = |[a_{i+k'}, b_{i-1}]| > |[d, R(d))| = r(d)$ because $|[a_{i+k'}, d]| \geq |[a_{i+k'}, b_{i+k'}]| > |(b_{i-1}, a_i)| \geq |(b_{i-1}, R(d))|$.

This contradicts our choice of d. The above contradiction applies even when $R(d) = a_i$, because in that case $d \rightarrow a_i$ and $a_i \rightarrow b_{i+k'}$, which implies that $(b_{i+k'}, d) \neq \emptyset$ (as $deg(a_i) \leq n-2$). If $R(d) \in (a_i, c)$, then there is a modification of C obtained from C by replacing $(R(d), c)$ with $[d, R(c)]$, contrary to the maximality of C. Therefore $j \in \{p+k'+1, p+k'+2, \ldots, i+k'\}$. Applying similar arguments, we have $r(d) = r(a_j)$, $d \in (b_{j-1}, b_j)$, and $R(d) \in (a_{j+k'}, a_{j+k'+1})$ (note that these arguments hold even when $j = i + k'$ because $a_{i+k'} \in (R(c), b_{i+k'-1})$. Finally we claim that a modification of K_m can be obtained from K_m by replacing $(R(d), c)$ with $[d, R(c)]$. Indeed, it follows from the above that $c \in (a_p, b_r), R(c) \in (b_{p+k'}, a_{r+k'+1}), d \in (b_{p+k'}, R(c))$, and $R(d) \in (a_p, c)$ (note that $c \rightarrow d$). Since $r(a_i) = r(c)$ and $r(a_j) = r(d)$, we have $|[a_i, c)| = |(b_{i+k'}, R(c)]|$ and $|[d, a_j)| = |(R(d), b_{j+k'}]|$. Therefore, $|[d, R(c)]| = |[a_i, b_{i+k'}]| + |[d, a_j)| + |(b_{i+k'}, R(c)]| > |(b_{j+k'}, a_i)| + |(R(d), b_{j+k'}]| + |[a_i, c)| = |(R(d), c)|$. (In these calculations we have assumed that $[d, R(c)] \supset [a_j, b_{i+k'}]$; otherwise we need to replace "$+|[d, a_j)|$" by "$-|[a_j, d)|$" and "$+|(R(d), b_{j+k'}]|$" by "$-|(b_{j+k'}, R(d)]|$" if $d \in (a_j, b_{i+k'})$ and similarly for $c \in (b_{j+k'}, a_i)$.) Furthermore, this modification is localized because of the choice of c and d. □

We observe here that the theorem implies that a localized modification exists if and only if a modification exists.

Consider now the following algorithm.

Algorithm 5.1.9 *Let D be a connected local transitive tournament with a round enumeration and with $\Delta(D) < n - 1$.*

Initialize $m \leftarrow -1$, $K_{-1} \leftarrow \emptyset$.

While K_m admits a modification, let K_{m+2} be a localized modification of K_m and increase m by 2.

Theorem 5.1.10 *We have*

- *Algorithm 5.1.9 correctly finds a maximum clique in D,*
- *Algorithm 5.1.9 can be implemented to run in time $O(n \log n)$, and*

- *there is an $O(m+n)$ algorithm to find a maximum clique in any proper circular arc graph.*

Proof: The algorithm will, in its first iteration, find a largest 1-overlap clique K_1. Clearly K_1 is localized, and, by Lemma 5.1.5, it is also admissible. Then Theorem 5.1.7 guarantees that all subsequent m-overlap cliques K_m are also admissible. At termination, K_m does not admit a modification, hence K_m is a maximum clique by Theorem 5.1.8.

We now discuss the implementation of Algorithm 5.1.9. Suppose that we have a local transitive tournament D with a round enumeration, i.e., suppose that we have the parameters $L(x)$, $R(x)$ (and hence $r(x)$) for each vertex x. In preparation, we can find in time $O(n)$ a vertex a with $r(a) = max\{r(x) : x \in V(D)\}$. Next we store the values $r(x)$ for $x \in [L(a), R(a)]$ in the leaves of a balanced tree structure, such as a $2-3$-tree [1], where each internal node stores the maximum value of $r(x)$ among its descendants. (To facilitate the calculation, we may in fact store in each internal node two values, the maximum $l(x)$ in its left subtree and the maximum $l'(x)$ in its right subtree.) This can be done in time $O(n \log n)$, [1]. Then, given any x and y, the tree can be pruned down, in time $O(\log n)$, to a subtree representing only the leaves between x and y, and hence having the value $M(x,y)$ stored in the root. This is explained in detail in [1], Section 4.12. Thus each evaluation of $M(x,y)$ (or $M[x,y], M[x,y)$, for which the computation is similar) takes time $O(\log n)$.

In each iteration we have a *current clique* K_m. We have noted above how to obtain the current clique K_1 of the first iteration. We shall maintain additional information which will allow us to estimate the complexity of the algorithm, as well as to proceed from K_m to K_{m+2} in the m-th iteration. (Note that we have first, third, fifth, etc. iterations and no second, fourth, etc. iterations, in this terminology.) Specifically, we shall *charge* certain vertices of D. The intention is to have the number of charged vertices proportional to $\log n$ times the work performed so far. A vertex will be charged at most once. We only charge vertices of the current clique. A charged vertex may be absent from later current cliques, but if it is not in K_m it will not be in K_{m+2} either.

Initially no vertex is charged. A defining interval $[a_i, b_i]$ of the current clique is *active* if not all of its vertices have been charged. We will operate on active intervals only.

In the first iteration we have $K_1 = [a, R(a)]$, no vertex has been charged and the unique defining interval $[a, R(a)]$ is active. In general we shall have the current clique K_m (which shall always be a subset of $[a, R(a)] \cup [L(a), a]$), some charged vertices and some active intervals. An ctive interval $[a_i, b_i]$ will either contain no charged vertices, or will consist of an interval $[a_i, f)$ of charged vertices and an interval $[f, b_i]$ of uncharged vertices. If there is an active interval of the second kind, there will be only one such interval and we will operate on it. (If all active intervals are of the first kind then we operate on any of them.)

To operate on an active interval $[a_i, b_i]$ which contains no charged vertices, we search, in the order from a_i to b_i, for the first vertex $f \in (a_i, b_i)$ such that for any $g \in (b_{i+m'}, R(f))$ with $r(g) = M(b_{i+m'}, R(f))$ we have

- $R(f) \in (b_{i+m'}, a_{i+m'+1})$,
- $R(g) \in (a_i, f)$, and
- $|[g, R(f)]| > |(R(g), f)|$.

If there is no such vertex f, then we charge all vertices of $[a_i, b_i]$. If there is such a vertex f, then we only charge all vertices of $[a_i, f)$.

To operate on an active interval $[a_i, b_i]$ in which the vertices ot $[a_i, f)$ are charged (and the vertices of $[f, b_i]$ uncharged), we perform the following operations:

- Find any vertices c and d such that $r(c) = M[f, b_i)$ and $r(d) = M(b_{i+m'}, R(c))$, and
- Define K_{m+2} to be obtained from K_m by replacing $(R(d), c)$ with $[d, R(c)]$, i.e., the defining intervals of K_{m+2} are all the defining intervals of K_m except $[a_i, b_i]$, plus the intervals $[a_i, R(d)], [c, b_i]$ and $[d, R(c)]$.

The correctness and the claimed complexity of our implementation will follow from the following observations, all of which have been asserted above.

1. If x is a vertex charged in the l-th iteration, and if $x \in K_m$ with $m > l$, then $x \in K_m, K_{m-2}, \ldots, K_l$.

2. An active interval $[a_j, b_j]$ contains no charged vertices or consists of an interval $[a_j, f)$ of charged vertices and an interval $[f, b_j]$ of uncharged vertices, for some $f \in (a_j, b_j)$. Moreover, there is always at most one active interval of the second kind.

3. A vertex is charged at most once.

4. The work performed in each iteration is proportional to $\log n$ times the number of vertices that have been charged during that iteration.

5. The clique K_{m+2} is a localized modification of the clique K_m.

Suppose that x is a vertex charged in the l-th iteration and let $x \in K_m$ with $m > l$. We shall show that $x \in K_{m-2}$. Let $[a_1, b_1], \ldots, [a_m, b_m]$ be the defining intervals of K_m. Say $x \in [a_j, b_j]$. If $x \notin K_{m-2}$, then $x \in (b_{j-1}, a_{j+1})$. In the l-th iteration, x belongs to an active interval of K_l. That active interval must be some $[a_i, b_k]$ such that $[a_i, b_k] \supset (b_{j-1}, a_{j+1})$ (since $m - 2 \geq l, a_j \neq a_i$). Recall that x was charged when we found (or failed to find) the first vertex $f \in (a_i, b_k)$ that satisfied the above conditions. But it is easy to see that the vertex a_j satisfies these conditions. Hence f must be in $(a_i, a_j]$ and so x would not have been charged. Thus $x \in K_{m-2}$. Now 1 follows.

Suppose $x \in [a_j, b_j]$ is a charged vertex. When x was charged, in the l-th iteration, $l \leq m$, we had $x \in [a_i, b_k]$ such that $[a_i, b_k] \supseteq [a_j, b_j]$. As above, we were searching for the first vertex $f \in (a_i, b_k)$ that satisfied the above conditions. If there was no f, or if $f \in (b_j, b_k)$, then each vertex of the entire interval $[a_j, b_j]$ has been charged. Otherwise, since x is charged, we must have $f \in (x, b_j]$, in which case $[a_j, b_j]$ consists of the interval $[a_j, f)$ of charged vertices and the interval $[f, b_j]$ of uncharged vertices. There is always at most one interval of this kind, because we keep processing it (without creating additional intervals of this kind), until there are only active intervals without charged vertices. This proves 2.

When a vertex is charged, we compute a value of $M(x,y)$ (or $M[x,y], M[x,y)$), at a cost of $O(\log n)$ (see above). This explains 4.

The assertion 5 follows from the fact that our construction of K_{m+2} agrees with that explained in the proof of Theorem 5.1.2.

To obtain an $O(m+n)$ algorithm for the maximum clique problem in proper circular arc graphs G we proceed as follows: First applying Algorithm 4.3.1 we can obtain in time $O(m+n)$ a local-transitive-tournament orientation D of G; secondly we use the method of Theorem 2.2.6 to find in time $O(m+n)$ a round enumeration of D. Hence we have parameters $R(x), r(x), L(x)$ for each vertex x of D and we can find in time $O(n)$ a vertex a with $r(a) = max\{r(x) : x \in V(D)\}$. Now instead of building a $2-3$-tree as above, we compute $M(x,y), M[x,y), M[x,y]$ for all pairs x, y such that $[x,y] \subseteq [a, R(a)]$ or $[x,y] \subseteq [L(a), a]$. This can be done in time $O(m+n)$ by dynamic programming, because $[a, R(a)]$ and $[L(a), a)$ are vertex-disjoint complete subgraphs of D and there are only $O(m)$ pairs of vertices x, y in $[a, R(a)]$ and in $[L(a), a]$. The iterations of Algorithm 5.1.9 can be done as above. Note that the work in each iteration of Algorithm 5.1.9 is now proportional to the number of vertices that have been charged during that iteration. □

5.2 The c-Colouring Algorithm

Assume again that D is a connected local transitive tournament (such that $\Delta(D) \leq n-2$), with a round enumeration $v_1, v_2, \ldots, v_n$. Let c be a fixed integer. In this section, we present an algorithm to decide whether or not D is c-colourable. We begin with the following two results of Orlin, Bonuccelli, and Bovet, reformulated from [61].

Lemma 5.2.1 *Suppose that n is divisible by c. Then D is c-colourable if and only there is no 1-overlap clique of size $c+1$. If there is no 1-overlap clique of size $c+1$, then the vertices of D can be coloured in clockwise circular order of the round enumeration, $1, 2, \ldots, c, 1, 2, \ldots, c, \ldots, 1, 2, \ldots, c$.* □

In general, we let $n = qc + r$, where $0 \le r < c$.

Lemma 5.2.2 *If D is c-colourable, then it can be coloured with c colours in such a way that r colour classes have $q + 1$ vertices each, and the remaining $(c - r)$ colour classes have q vertices each.* □

Consider now the following algorithm.

Algorithm 5.2.3 *Let D be a connected local transitive tournament with a round enumeration and with $\Delta(D) < n - 1$.*

Step 1. Find a maximum clique C which is a k-overlap clique with $|C| = \omega$.

Step 2. If $\omega > c$, then D is not c-colourable.

Step 3. If $\omega < c$ and $n > (c-1)^2$, then D is c-colourable by the technique explained in Lemma 5.2.4.

Step 4. If $\omega < c$ and $n \le (c-1)^2$, then determine whether or not D is c-colourable by the algorithm from [70].

Step 5. If $\omega = c$ and $r = 0$, then D is c-colourable by the method explained in Lemma 5.2.1;

Step 6. If $\omega = c$ and $r \ge 1$, then determine whether or not D is c-colourable by the technique of Lemma 5.2.5 if $k > 1$, and of Lemma 5.2.6 if $k = 1$.

Step 1 of our Algorithm 5.2.3 can be done in time $O(m+n)$ according to Corollary 5.1.10. Step 2 can be done in time $O(1)$. For Step 3 we apply the technique (easily implemented in time $O(n)$) inherent in the following lemma.

Lemma 5.2.4 *If $\omega < c$ and $n > (c-1)^2$, then D is c-colourable.*

Proof: Since $\omega \leq c-1$ we can construct a colouring in which consecutive vertices in clockwise circular order of the round enumeration obtain colours $1, 2, \ldots, c, 1, 2, \ldots, c$ as well as $1, 2, \ldots, (c-1), 1, 2, \ldots, (c-1)$, provided we can fit these "runs" together to yield n. Since $n > (c-1)^2$, this can for example be done as follows: Let $n = p(c-1) + s$ where $0 \leq s < c-1$. Since $n > (c-1)^2$, we have $p \geq (c-1)$. Colour the first sc vertices in clockwise circular order of the round enumeration by $1, 2, \ldots, c, 1, 2, \ldots, c, \ldots, 1, 2, \ldots, c$, and colour the remaining $(p-s)(c-1)$ vertices by $1, 2, \ldots, (c-1), 1, 2, \ldots, (c-1), \ldots, 1, 2, \ldots, (c-1)$. To see this is a proper c -colouring, suppose that there are two adjacent vertices v_i and v_j which obtain the same colour. Since v_i and $_j$ are adjacent, we know from the definition of a round enumeration that either $[v_i, v_j]$ or $[v_j, v_i]$ is complete. But each of $[v_i, v_j]$ and $[v_j, v_i]$ has size at least c, contradicting the hypothesis that $\omega < c$. □

Step 4 takes time $O(1)$ since c is fixed and so $O(n^{3/2}) = O(c^3) = O(1)$. Step 5 can be easily executed in time $O(n)$ according to Lemma 5.2.1.

For Step 6, suppose first that $\omega = c$, $r > 0$, and $k \geq 3$. Let $C = [a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ be the maximum clique of D found in Step 1.

Let $i = 1, 2, \ldots,$ or k, and suppose that $|[a_i, b_i]| = l$ and $|(b_{i+k'}, a_{i+k'+1})| = s$. Let $y_1, y_2, \ldots, y_s$ be the vertices of $(b_{i+k'}, a_{i+k'+1})$ listed in clockwise circular order of the round enumeration. Note that $l > s$ by the definition of a k-overlap clique. Let H_i be the complement of the underlying graph of the subgraph of D induced by $[a_i, b_i] \cup (b_{i+k'}, a_{i+k'+1})$. Since each of $[a_i, b_i]$ and $(b_{i+k'}, a_{i+k'+1})$ induces a complete subgraph of D, the graph H_i is bipartite. We shall say that H_i has a *round matching* if there exist vertices $x_1, x_2, \ldots, x_s$ of $[a_i, b_i]$, appearing in clockwise circular order, such that $y_1x_1, y_2x_2, \ldots,$ and y_sx_s are edges of H_i. Note that it is easy to determine in time $O(|H_i|)$ whether or not each H_i has a round matching.

Lemma 5.2.5 *The digraph D is c-colourable if and only if each graph H_i $(i = 1, 2, \ldots, k)$ has a round matching.*

Proof: Suppose that H_i has the round matching $y_1x_1, y_2x_2, \ldots, y_sx_s$. Then D can be coloured with c colours in the following way: Arbitrarily colour the vertices in

$[a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ with c colours. Then colour each vertex ($j = 1, 2, \ldots, s$) $y_j \in (b_{i+l'}, a_{i+l'+1})$ with the colour assigned for the matched vertex x_j. It is easy to verify that this is a proper c -colouring of D.

Suppose in turn that D is c-coloured. Note that the vertices in $[a_1, b_1] \cup [a_2, b_2] \cup \ldots \cup [a_k, b_k]$ must obtain c distinct colours. Moreover, vertices in $(b_{i+k'}, a_{i+k'+1})$ may use only colours assigned to vertices in $[a_i, b_i]$ since every vertex in $(b_{i+k'}, a_{i+k'+1})$ is adjacent to every vertex in each $[a_j, b_j]$ for $j \neq i$. Hence H_i must have a matching of size s. We apply induction on s to show it must also have a round matching of size s. When $s = 1$, there is nothing to prove. When $s > 1$, let $M = \{y_1x'_1, y_2x'_2, \ldots, y_sx'_s\}$ be a matching in H_i; let $\{x_1, x_2, \ldots, x_s\} = \{x'_1, x'_2, \ldots, x'_s\}$, where $x_1, x_2, \ldots, x_s$ are listed in clockwise circular order. If $x'_1 = x_1$, then we are done by induction. Otherwise $x'_1 = x_f$ and $x_1 = x'_g$, for some $f > 1$ and $g > 1$. Thus y_1x_f and y_gx_1 are edges of H_i, and hence y_1 is not adjacent to x_f, and x_1 is not adjacent to y_g in D. If y_1 is adjacent to x_1 in D, then either $[y_1, x_1]$ induces a complete subgraph and it implies that y_g and x_1 are adjacent, or $[x_1, y_1]$ induces a complete subgraph and it implies that y_1 and x_f are adjacent in D, contradicting the hypothesis. Therefore y_1 is not adjacent to x_1. A similar argument applies to show that y_g is not adjacent to x_f. Consequently, $\{y_1x'_g, y_2x'_2, \ldots, y_gx'_1, y_{g+1}x'_{g+1}, \ldots, y_sx'_s\}$ is a matching of H_i in which $x'_g = x_1$. Therefore, by induction H_i has a matching of the required form. □

By Lemma 5.2.5 the first part of Step 6 can be implemented in time $O(n)$.

Finally suppose that $\omega = c$, $r > 1$, and $k = 1$. Let C be a maximum clique of D which is a 1-overlap clique. Without loss of generality, assume that $C = \{v_1, v_2, \ldots, v_c\}$. We need to decide whether or not there exists a c-colouring of D; according to Lemma 5.2.2 it is enough to seek a c-colouring with r 'larger' classes of size $q + 1$ and $c - r$ 'smaller' classes of size q. The clique C must have one vertex from each of the larger colour classes. In other words, if there exists a c-colouring of D, then for some c-colouring of D and some set Y of r vertices of C, it is the case that $D - Y$ has precisely q vertices of each of c colours. Since c is fixed, there is only a constant number C_r^c of possible sets Y. Thus we may fix a set Y of r vertices of C, $y_1^0, y_2^0, \ldots, y_r^0$ (listed in clockwise circular order), and ask if there exists a c-colouring of D in which $D - Y$ has precisely q vertices of each of the c colours. Note that we only need to colour the vertices of $D - C$. We have a linear order on $D - C$, induced by clockwise order of D. We shall in the sequel use terms like "before, after, precedes, follows, first, last, next", etc., with reference to this linear order, $v_{c+1}, \ldots, v_n$.

For each vertex y_i^0, we shall associate a set of "stretch" $B_i^0, B_i^1, \ldots, B_i^q$ in D, which will guide our choice of a c-colouring. Suppose $y_i^0 = v_s$. The stretch B_i^0 consists of the first r vertices starting from $y_i^0 = v_s$, i.e., $v_s, v_{s+1}, \ldots, v_{s+r-1}$, and each subsequent stretch B_i^j consists of the next c consecutive vertices (in clockwise circular order). Thus for $1 \leq j \leq q$,

$$B_i^j = \{v_{(j-1)c+s+r}, v_{(j-1)c+s+r+1}, \ldots, v_{jc+s+r-1}\}.$$

For convenience, we use f_i^j to denote the first vertex of B_i^j and l_i^j the last vertex of B_i^j with respect to clockwise circular order of the round enumeration.

We will seek a c-colouring in a greedy fashion, guided by the associated stretches. Specifically, we shall find vertices

$$y_1^1, y_2^1, \ldots, y_r^1, y_1^2, y_2^2, \ldots, y_r^2, \ldots, y_1^q, y_2^q, \ldots, y_r^q$$

in such a way that y_i^j is the first vertex of B_i^j which is not dominated by y_i^{j-1} and comes after y_{i-1}^j (or after y_r^{j-1} if $i = 1$). The existence of such a sequence will follow from Lemma 5.2.6.

Let $C_i = \{y_i^0, y_i^1, \ldots, y_i^q\}$, with $1 \leq i \leq r$. We show that each class C_i is either independent or contains the single arc $y_i^q y_i^0$. Suppose y_i^k dominates y_i^j. If $k < j$ then y_i^k dominates y_i^{k+1} contrary to the choice of y_i^{k+1}. If $j < k < q$, then again y_i^k dominates y_i^{k+1}, a contradiction. Finally, if $0 < j < k$, then y_i^{j-1} dominates y_i^j, again a contradiction. Thus the only possible arc inside the set C_i is $y_i^q y_i^0$.

Suppose that all classes C_i are independent and define $D' = D - C_1 - C_2 - \ldots - C_r$. We shall show below that D' contains no 1-overlap clique of size $c - r + 1$. Therefore it can, by Lemma 5.2.1, be coloured with $c - r$ colours forming the colour classes $C_{r+1}, C_{r+2}, \ldots, C_c$. Clearly, $C_1, C_2, \ldots, C_c$ is a c-colouring of D.

Otherwise we shall try a different set Y. We shall prove below that if no set Y allows a desired colouring, then D is not c-colourable.

Lemma 5.2.6 *If D is c-colourable, then there exists a set Y of r vertices of C such that D has a c-colouring in which $D - Y$ has precisely q vertices of each of the c colours.*

Proof: Suppose that D is c-colourable. Then by Lemma 5.2.2 there is a c-colouring of D with r colour classes of size $q+1$. Suppose that the vertices of these r colour classes are

$$x_1^0, x_2^0, \ldots, x_r^0, x_1^1, x_2^1, \ldots, x_r^1, \ldots, x_1^q, x_2^q, \ldots, x_r^q,$$

listed in clockwise circular order. Applying Lemma 5.2.1 to the subgraph induced by the x_i^j, we see easily that each $D_i = \{x_i^0, x_i^1, \ldots, x_i^q\}$ is a colour class of the above c-colouring of D.

Note that C contains exactly one vertex of each D_i, $i = 1, 2, \ldots, r$. Without loss of generality, we may assume that (for each $i = 1, \ldots, r$) $x_i^0 \in C$. Let $y_i^0 = x_i^0$, i.e., let $Y = \{x_1^0, x_2^0, \ldots, x_q^0\}$. Suppose that

$$y_1^1, y_2^1, \ldots, y_r^1, y_1^2, y_2^2, \ldots, y_r^2, \ldots, y_1^q, y_2^q, \ldots, y_r^q$$

is the sequence of vertices defined above.

This sequence is well defined. Indeed, suppose that

$$y_1^1, y_2^1, \ldots, y_r^1, y_1^2, y_2^2, \ldots, y_r^2, \ldots, y_1^{j-1}, y_2^{j-1}, \ldots, y_r^{j-1}$$

have been found as required. There is in each B_i^j $(i = 1, \ldots, r)$ a vertex which is not dominated by y_i^{j-1} and which comes after y_{i-1}^j (or y_r^{j-1} if $i = 1$). In fact, if $y_i^{j-1} = v_s$ then v_{s+c} is such a vertex. In particular, $v_{s+c} = y_i^j \in B_i^j$ because $v_s = y_i^{j-1} \in B_i^{j-1}$, and y_i^j comes after y_{i-1}^j (or y_r^{j-1} if $i = 1$) because y_i^{j-1} is after y_{i-1}^{j-1} (or y_r^{j-2} if $i = 1$). This also implies that y_i^j comes at most $c+1$ vertices after y_i^{j-1}. Therefore, after we remove all y_i^j $(i = 1, \ldots, r, j = 1, \ldots, q)$ from D, there is no 1-overlap clique of size $c - r + 1$. We also note that by definition

$$y_1^1, y_2^1, \ldots, y_r^1, y_1^2, y_2^2, \ldots, y_r^2, \ldots, y_1^q, y_2^q, \ldots, y_r^q$$

appear in clockwise circular order.

Claim. For each i and j, we have $y_i^j \in (y_i^0, x_i^j]$ or $y_i^j = f_i^j$.

It is not difficult to see that the Claim holds for $j = 1$.

Suppose to the contrary that there is some y_i^j ($j > 1$) with $y_i^j \neq f_i^j$ and $y_i^j \notin (y_i^0, x_i^j]$. We may assume y_i^j is the first such vertex in clockwise circular order. Then $x_i^j \notin [l_i^j, y_i^0)$ and hence $x_i^j \in (y_i^0, l_i^j)$. Consider the vertex y_i^{j-1}. Then $y_i^{j-1} \in (y_i^0, x_i^{j-1}]$ or $y_i^{j-1} = f_i^{j-1}$.

If $y_i^{j-1} = f_i^{j-1}$, then $y_i^j = f_i^j$, by the definition of y_i^j, contradicting our hypothesis. Suppose that $y_i^{j-1} \in (y_i^0, x_i^{j-1}]$. Then $y_i^{j-1}, x_i^{j-1}, x_i^j, y_i^j$ are in clockwise circular order. Hence y_i^{j-1} does not dominate x_i^j because x_i^{j-1} is not adjacent to x_i^j. We consider the following two cases.

Case 1. If $x_i^j \in [f_i^j, l_i^j)$, then at least one vertex from $y_{i+1}^{j-1}, \ldots, y_r^{j-1}, y_1^j, \ldots, y_{i-1}^j$ must be in $[x_i^j, l_i^j)$, as otherwise $y_i^j \in (f_i^j, x_i^j]$, contradicting our hypothesis. Let y_a^b be such a vertex. Then $y_a^b \in [f_i^j, l_i^j)$. This implies that $y_a^b \neq f_a^b$ (because the stretch B_a^b precedes the stretch B_i^j) and hence $y_a^b \in (y_a^0, x_a^b]$. Therefore x_i^j precedes (or equals) y_a^b, which precedes (or equals) x_a^b. This means that x_i^j precedes x_a^b while y_i^j follows y_a^b, which is impossible according to our numbering conventions.

Case 2. If $x_i^j \in (y_i^0, f_i^j)$, then $y_i^{j-1}, x_i^{j-1}, x_i^j, f_i^j$ appear in this order. Hence y_i^{j-1} does not dominate f_i^j because x_i^{j-1} and x_i^j are not adjacent. Since $y_i^j \neq f_i^j$, there exists a vertex y_a^b among $y_{i+1}^{j-1}, \ldots, y_r^{j-1}, y_1^j, \ldots, y_{i-1}^j$ which is after f_i^j. This implies that y_a^b must be after x_i^j and thai y_a^b can not be the first vertex f_a^b of B_a^b. So y_a^b must precede (or equal) x_a^b. Therefore x_i^j precedes x_a^b while y_i^j follows y_a^b, which violates our numbering conventions.

Applying the Claim to y_i^q, we conclude that either $y_i^q \in (y_i^0, x_i^q]$ or $y_i^q = f_i^q$ for each $i = 1, 2, \ldots, r$. Suppose that $y_i^q \in (y_i^0, x_i^q]$. Then y_i^q does not dominate y_i^0 because $[y_i^q, y_i^0] \supset [x_i^q, x_i^0]$ and x_i^q is not adjacent to x_i^0. Suppose that $y_i^q = f_i^q$. Then y_i^q does not dominate y_i^0 because $|[y_i^q, y_i^0]| > c$. Hence the only possible arc $y_i^q y_i^0$ contained in C_i does not exist, namely, the set $C_i = \{y_i^0, y_i^1, \ldots, y_i^q\}$ is independent.

The digraph $D - C_1 - \ldots - C_r$ has $q(c-r)$ vertices and no 1-overlap clique of size $c - r + 1$. Thus by Lemma 5.2.1 it has a $(c-r)$-colouring in which each colour class has q vertices. □

Note that all procedures in the Lemma take $O(n)$ time.

Theorem 5.2.7 *Let c be fixed.*

- *Algorithm 5.2.3 correctly decides whether or not D is c-colourable.*
- *Step 1 of Algorithm 5.2.3 can be implemented to run in time $O(m+n)$, and in time $O(n \log n)$ if a proper circular arc representation is given.*
- *The remaining steps of Algorithm 5.2.3 can be implemented to run in time $O(n)$.*
- *There is an $O(m+n)$ algorithm to decide whether or not a proper circular arc graph is c-colourable and there is an $O(n \log n)$ algorithm to decide whether or not a proper circular arc graph is c-colourable if a proper circular arc representation is given.* □

Chapter 6

In-tournaments

6.1 Introduction

An oriented graph D is an *in-tournament* if the inset of each vertex induces a tournament. If the outset of each vertex of D induces a tournament, then D is an *out-tournament*. It is easy to see that a full reversal of an in-tournament is an out-tournament; similarly a full reversal of an out-tournament is an in-tournament. For this reason, we only deal with in-tournaments as all results are transferable to out-tournaments. A local tournament is of course an in-tournament and an out-tournament. So the class of in-tournaments properly contains the class of local tournaments. Note that any induced subdigraph of an in-tournament is an in-tournament.

We have seen that many nice properties of tournaments remain valid for local tournaments. In the first part of this chapter we will investigate which of these properties hold also for in-tournaments. As we shall see in Section 6.2 it turns out that in-tournaments still have considerable structure. It follows easily from the results given in this section that deciding whether an in-tournament has a hamiltonian path, a hamiltonian cycle, or a cycle through two given vertices x and y can all be done in polynomial time.

The second motivation of studying in-tournaments was an open problem due to Skrien [71]: Using our notation it is the problem of characterizing those graphs which are orientable as in-tournaments. For local tournaments, the analogous question was treated in [39] and [71]. We are not able to give a complete characterization in terms of forbidden induced subgraphs. However we prove that chordal graphs and graphs representable (cf. Section 6.3) in unicyclic graphs are orientable as in-tournaments. We also characterize those line graphs that can be oriented as in-tournaments. In the final section of this chapter, we briefly discuss orientations of graphs as strong in-tournaments. We give examples of classes of graphs that can be oriented in this way, as well as an example of a class of graphs which are orientable as in-tournaments precisely when they are orientable as strong in-tournaments.

An *in-semicomplete digraph* is a digraph in which the inset of each vertex induces a semicomplete digraph; similarly an *out-semicomplete digraph* is a digraph in which the outset of each vertex induces a semicomplete digraph. *In-tournaments* and *out-tournaments* are defined as above for oriented graphs. So a locally semicomplete digraph is a digraph which is both in-semicomplete and out-semicomplete.

An *in-branching* is a spanning tree rooted at some vertex v and oriented in such a way that every vertex other than v has one arc out of it. An *out-branching* is defined analogously. For any positive integer k, the *k-th power* D^k of a digraph D has the same vertex set as D, and a vertex x dominates a vertex y whenever there is a directed (x, y)-path of length at most k in D [5].

We close this section by giving a characterization of in-semicomplete digraphs that will be of use in Section 6.3. A *pointed set* is a pair consisting of a set and one element in it. The *catch digraph* [54] $\Omega(F)$ of a family $F = ((S_x, p_x)/x \in V)$ of pointed sets has vertex set V and an arc from x to y if $p_y \in S_x$, for $x \neq y \in V$. The *intersection graph* $\Gamma(F')$ of a family $F' = (S_x/x \in V)$ of sets has vertex set V and two distinct vertices x and y are adjacent whenever $S_x \cap S_y \neq \emptyset$. Obviously the underlying graph of $\Omega((S_x, p_x)/x \in V)$ is a spanning subgraph of $\Gamma(S_x/x \in V)$ for any family of pointed sets. The converse does not hold in general. However we have the next result.

Lemma 6.1.1 *If D is an in-semicomplete digraph, then $\Omega((O[x], x)/x \in V) = D$ and $\Gamma(O[x]/x \in V) = G(D)$.*

Proof: The first statement is obvious. Now let x, y be distinct vertices of D such that $O[x] \cap O[y] \neq \emptyset$. Then $x \rightarrow y$ or $y \rightarrow x$ or x and y have some common successor z. In the latter case, either $x \rightarrow y$ or $y \rightarrow x$, since D is in-semicomplete. Then $G(D) = \Gamma(O[x]/x \in V)$ by the remarks above. □

Theorem 6.1.2 *[84]: A digraph $D = (V, A)$ is in-semicomplete if and only if it is the catch digraph of a family $((S_x, p_x)/x \in V)$ such that $G(D)$ equals $\Gamma(S_x/x \in V)$.*

Proof: Let D be the catch digraph of $((S_x, p_x)/x \in V)$ such that $G(D)$ is the intersection graph G of $(S_x/x \in V)$. Choose any predecessors x and z of a vertex y. Then $p_y \in S_x \cap S_z$, which implies $xz \in E(G)$. But then $x \rightarrow z$ or $z \rightarrow x$ in D. The converse follows from Lemma 6.1.1. □

6.2 On the Structure of In-tournaments

In this section we study the properties of in-tournaments and show that some of the basic and very nice properties of tournaments extend not only to local tournaments, but even to this more general class of digraphs.

6.2.1 Path Merging in In-tournaments

The first result is a very useful property of in-tournaments. We say that a digraph is *in-path-mergeable* if it has the property that for any choice P_1, P_2 of internally vertex-disjoint paths with terminal vertices x, z and y, z, respectively, there exists a path P with initial vertex x or y and terminal vertex z such that $V(P) = V(P_1) \cup V(P_2)$ and vertices from the same path P_i ($i = 1$ or 2) remain in the same order in P. The path P is called *the merging* of P_1 and P_2.

Proposition 6.2.1 *In-tournaments are in-path-mergeable and the merging can be done in $O(m + n)$ time.*

Proof: Let P_1 and P_2 be internally vertex-disjoint (x,z)– and (y,z)-paths, respectively. Let p_1 and p_2 denote the lengths of these paths. We shall prove the first claim by induction on $p_1 + p_2$. The case $p_1 + p_2 = 2$ is trivial, so assume $p_1 + p_2 \geq 3$. Let z_1 and z_2 denote the predecessor of z on P_1 and P_2, respectively. By the definition of an in-tournament, z_1 and z_2 are adjacent. Assume without loss of generality that $z_1 \rightarrow z_2$. If $z_2 = y$, then $P = P_1[x, z_1] \cup \{z_1 \rightarrow y \rightarrow z\}$ is the desired path. Otherwise apply induction to the paths $P_1[x, z_1] \cup \{z_1 \rightarrow z_2\}$ and $P_2[y, z_2]$. The proof is easily turned into a $O(m+n)$ time algorithm. □

Corollary 6.2.2 *Let D be an in-tournament with two distinct vertices x and y, such that there are two internally vertex-disjoint (x,y)-paths P_1 and P_2 in D. Then P_1 and P_2 can be merged into one (x,y)-path P such that $V(P) = V(P_1) \cup V(P_2)$. Such a path P can be found in $O(m+n)$ time.* □

One will often use Corollary 6.2.2 in the following form.

Corollary 6.2.3 *Let $P_1 = x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_p$ and $P_2 = y_1 \rightarrow y_2 \rightarrow \ldots \rightarrow y_q$ be disjoint paths in an in-tournament D. If there exist $i, j, 1 \leq i < j \leq p$, such that $x_i \rightarrow y_1, y_q \rightarrow x_j$ then D has an (x_1, x_p)-path P such that $V(P) = V(P_1) \cup V(P_2)$.*

Proof: Apply Corollary 6.2.2 to the paths $P_1[x_i, x_j]$ and $x_i \rightarrow y_1 \rightarrow \ldots \rightarrow y_q \rightarrow x_j$. □

The proof of the next result shows the usefulness of the merging property. For any pair of vertices x and y we use $d(x,y)$ to denote the length of a shortest directed (x,y)-path in D, if there exists one.

Proposition 6.2.4 *Any power of an in-semicomplete digraph is in-semicomplete.*

Proof: Let D be an in-semicomplete digraph, and let D^k be the k-th power of D for an integer $k \geq 2$. We claim that for any three vertices x, y, z of D the following property holds: If $d(x,y) \leq k$ and $d(z,y) \leq k$, then $d(x,z) \leq k$ or $d(z,x) \leq k$. Clearly it is enough to prove the claim for internally vertex-disjoint $(x,y)-,(z,y)$-paths. Now it is easy to see that the claim follows from Proposition 6.2.1, since any two such internally vertex-disjoint paths can be merged such that the relative order of the vertices from the same path is retained. □

In [6] it is shown that any digraph D with the path-merging property - that is, for any two internally disjoint paths P_1 and P_2 with the same initial vertex x and the same terminal vertex y, there exists an (x,y)-path P, such that $V(P) = V(P_1) \cup V(P_2)$ - still has a hamiltonian cycle whenever it can possibly have one, i.e., whenever D is strong and $G(D)$ has no cutvertex. Furthermore this class of digraphs properly contains the class of in-tournaments.

6.2.2 The Strong Components of In-tournaments

Next we turn to the structure of the strong components of in-tournaments. For local tournaments, the structure is very similar to that of tournaments: Any strong component is a tournament; if there is an arc between two strong components, then one completely dominates the other; and finally $SC(D)$ has a unique spanning path (cf. Lemma 2.2.4). For in-tournaments, not all of this structure is retained. However, as we shall see there is still a lot of structure.

Lemma 6.2.5 *Every connected in-tournament has an out-branching.*

Proof: We use induction on n. If $n \leq 2$ this is clear, so assume $n \geq 3$. Let x be a vertex such that the underlying graph of $D - x$ is connected. We see that $D - x$ is an in-tournament and, by induction, it has an out-branching. If x is dominated by some vertex of $D - x$, then the claim follows. Hence we may assume that x dominates some vertex $y \in D - x$ and is not dominated by any vertex. Now, it follows from Proposition 6.2.1 that D has an out-branching rooted at x. □

Theorem 6.2.6 *Let D be an in-tournament.*

(a) *Let A and B be distinct strong components of D. If a vertex $a \in A$ dominates some vertex in B, then $a \rightarrow B$. Furthermore $A \cap I(b)$ induces a tournament for each $b \in B$.*

(b) *If D is connected, then $SC(D)$ has an out-branching. Furthermore, if R is the root and A is any other component, there is a path from R to A containing all the components that can reach A.*

Proof: Let A and B be strong components of D for which there is an arc $a \rightarrow b$ from A to B. Since B is strong, there is a (b', b)-path for any $b' \in B$. Hence, it follows from the definition of an in-tournament and the fact that there are no arcs from B to A that $a \rightarrow b'$. This proves the first part of (a). The second part of (a) is immediate from the definition of an in-tournament.

The first part of (b) follows by observing that $SC(D)$ is itself an in-tournament and then applying Lemma 6.2.5. The second part follows from Proposition 6.2.1. We leave the details to the reader. □

Let B and C be two vertex-disjoint connected subgraphs of a digraph D. A $B - C$ *separating set* is a subset $S \subset V(D)$ such that B and C are in distinct components of $D - S$. A $B - C$ separating set is *minimal* if B and C are in the same component of $D - S'$ for any $S' \subset S$. A *minimal separating set* of a strong digraph D is a subset $S \subset V(D)$ such that $D - S$ is not strong, but $D - S'$ is strong for any $S' \subset S$.

Corollary 6.2.7 *Let D be a strong in-tournament and let S be a minimal separating set. There is a unique order $D_1, D_2, \ldots, D_k$ of the strong components of $D - S$, such that there are no arcs from D_j to D_i for $j > i$ and D_i has an arc to D_{i+1} for $i = 1, \ldots, k-1$.*

Proof: We shall prove that $D - S$ has precisely one sink component. Suppose $D - S$ has at least two sink components. By the minimality of S every vertex $z \in S$ must be dominated by at least one vertex from each sink component of $D - S$. Thus by the definition of an in-tournament, all sink components are adjacent, contradicting the fact that they are sink components. Hence $D - S$ has precisely one sink component and the claim follows from Theorem 6.2.6 (b) (when there is only one sink component, every component has a directed path to that component). □

6.2.3 Paths and Cycles in In-tournaments

We begin this section by characterizing those in-tournaments that have hamiltonian paths. In [4] it was shown that every connected local tournament has a hamiltonian path. This does not extend to in-tournaments (e.g., take any out-branching with at least two branches), but as we shall see below, there is still a good characterization of those in-tournaments that have hamiltonian paths.

Theorem 6.2.8 *A connected in-tournament D has a hamiltonian path if and only if it has an in-branching.*

Proof: Since any hamiltonian path is an in-branching we need only prove the other half of the claim. Suppose D has an in-branching. Using Proposition 6.2.1 it is easy to prove, by induction on the number of branches of the in-branching, that D has a hamiltonian path ending in the root of the in-branching. We leave the details to the reader. □

Corollary 6.2.9 *There is a polynomial algorithm to decide if a given in-tournament has a hamiltonian path and find one if it exists.*

Proof: For any digraph D deciding the existence of an in-branching and finding one if it exists can be done in $O(m \log n)$ (see [75]). Given an in-branching of D, its branches can be merged into a hamiltonian path ending in x in time $O(n^2)$ by Proposition 6.2.1. □

Now we show that just as for tournaments and local tournaments, every strong in-tournament has a hamiltonian cycle. First we prove a result which has several nice consequences, as we shall see below.

Theorem 6.2.10 *Let D be a strong in-tournament having a cycle of length k, but no cycle of length $k+1$ $(k < n)$. Then it has cycles of all lengths $l+1, l+2, \ldots, l+k$ for some l with $2 \le l \le n-k$.*

Proof: Let C be a cycle of length k in D. Since $k < n$ and D is strong, there exits a vertex $x \in V(D) - V(C)$ such that x dominates a vertex on C. If x is also dominated by some vertex of C, then it follows from Corollary 6.2.2 that D has a cycle of length $k+1$. Hence we may assume that x is not dominated by any vertex of C. Now we conclude, by the fact that D is an in-tournament, that x dominates all of C. Since D is strong, there exists a directed path P from C to x, let l denote the length of P. Since $x \rightarrow C$, $l \ge 2$. Now, since $x \rightarrow C$, we conclude that D has directed cycles of lengths $l+1, \ldots, l+k$, all containing P as a subpath. □

Corollary 6.2.11 *An in-tournament D has a hamiltonian cycle if and only if it is strong. Furthermore there is a polynomial algorithm to find a hamiltonian cycle in any strong in-tournament.*

Proof: Since D is strong, it has a cycle. By Theorem 6.2.10, the length of a longest cycle must be n, so D is hamiltonian. It is easy to derive an $O(n^3)$ algorithm for finding a hamiltonian cycle from the proof of Theorem 6.2.10. We leave the details to the reader. □

Corollary 6.2.12 *Any two vertices in each strong component of an in-tournament lie on a cycle.*

Proof: This is immediate from Corollary 6.2.11. □

Corollary 6.2.13 *Let D be a strong in-tournament. If D has a cycle C of length k, for some $k \ge \lceil \frac{n}{2} \rceil$, then D has cycles of all lengths $k, k+1, \ldots, n$.*

Proof: This follows immediately from Theorem 6.2.10 by backwards induction on k. □

Corollary 6.2.14 *Any strong in-tournament D which is not a directed cycle contains a vertex x such that $D - x$ is strong.*

Proof: Let D be a strong in-tournament on n vertices which is not a directed cycle. It follows easily from Theorem 6.2.10 that D has a cycle of length k for some $\lceil \frac{n}{2} \rceil \leq k < n$. Thus the claim follows from Corollary 6.2.13. □

An oriented graph $D = (V, A)$ is *pancyclic* if it contains a directed cycle of length l for each $l = 3, 4, \ldots, |V|$.

Corollary 6.2.15 *An in-tournament D for which $G(D)$ is chordal is pancyclic if and only if it is strong.*

Proof: This follows from Corollary 6.2.11 and Corollary 6.2.14 by induction. □

Note that Corollary 6.2.13 cannot be extended to cycles of length $k, k+1, \ldots, n$ through some specific vertex, as was the case for local tournaments (see Theorem 3.4 in [4]). This is shown by the digraph D in Figure 6.1, where $r < k$. By Corollary 6.2.13, D has cycles of all lengths $k, k+1, \ldots, n$, but the vertex c_k is not on any cycle of length s with $k < s < n$. By choosing $r = k+1$, we get a family of digraphs showing that $k \geq \lceil \frac{n}{2} \rceil$ is best possible for Corollary 6.2.13. This digraph has a cycle of length $\lfloor \frac{n}{2} \rfloor$ but no cycle of length $\lceil \frac{n}{2} \rceil$.

Before closing this section we point out that all the results in Section 6.2 are true for in-semicomplete digraphs as well. We also point out that in [8] it is shown that by a more detailed inspection and use of suitable datastructures, one can obtain $O(m + n \log n)$ algorithms for finding hamiltonian paths and cycles in in-tournaments if they exist.

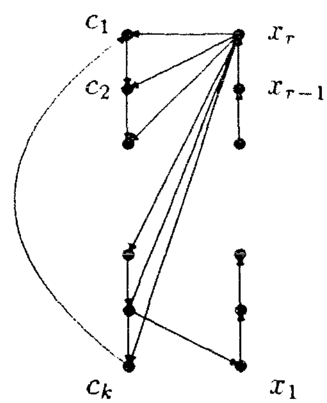

Figure 6.1: An in-tournament D on $k+r$ vertices, $r < k$, where the vertex c_k is not on an $s-$cycle for any $k < s < k+r$.

6.3 In-tournament Orientability

Theorem 6.3.1 *Graphs that are orientable as in-tournaments can be recognized in polynomial time.*

Proof: Let a graph $G = (V, E)$ be given, and let $A = \{a_1, a_2, \ldots, a_m\}$ be an arbitrary orientation of the edges of G . If a_i is an orientation of the edge yz of G, then the reverse orientation of that edge is denoted by $\overline{a}_i$. We now construct an instance of the 2-SAT problem as follows: The set of literals is $X = \{a_1, \ldots, a_m, \overline{a}_1, \ldots, \overline{a}_m\}$, and two such literals ℓ_i and ℓ_j lie in a common clause $(\ell_i \vee \ell_j)$ precisely when $\overline{\ell}_i, \overline{\ell}_j$ correspond to arcs with the same terminal vertex and non-adjacent initial vertices. It is easy to see that G is orientable as an in-tournament if and only if the above-defined instance of 2-SAT is satisfiable. The complexity of 2-SAT is $O(\#clauses)$ (see [62]). Hence, it follows from the way we construct the clauses above that we can recognize graphs orientable as in-tournaments in time $O(m\Delta)$, where Δ denotes the maximum degree of G. □

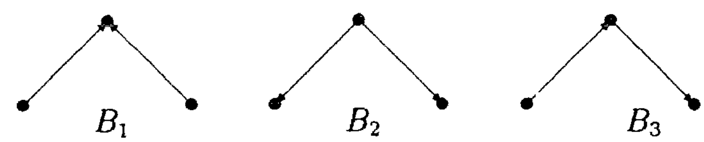

Figure 6.2: The digraphs B_1, B_2, B_3

Let $\mathcal{B}$ be the family of the three digraphs shown in Figure 6.2 and let F be any subset of $\mathcal{B}$ other than $\{B_1\}$ or $\{B_2\}$. Skrien [71] characterized the classes of those graphs which can be oriented without a member of F as an induced subdigraph. These are the classes of complete graphs, comparability graphs, proper circular arc graphs, and nested interval graphs [71]. Since each of the forbidden configurations contains just two arcs, 2-SAT could be used to solve the recognition problem for each of these four classes, all in time $O(m\Delta)$.

A graph G is called *representable* in the graph H if G is isomorphic to the intersection graph of a family of connected subgraphs $(H_x/x \in V(G))$ of H. It seems interesting that three of these four classes above can be defined by representability. In the case of the underlying graphs of in-tournaments, we have not been able to find a similar characterization. However, we have the following sufficient condition in terms of representability.

Theorem 6.3.2 *[64]: Every graph that is representable in a unicyclic graph is orientable as an in-tournament.*

Proof: Let $(H_x/x \in V(G))$ be a representation of G in the unicyclic graph H with cycle $C = z_0, z_1, \ldots, z_{\ell-1}$. The numbering is done clockwise around the cycle (the reader should think of this as drawn in the plane). We may assume H connected. For vertices x of G whose representative H_x contains all vertices of the cycle C, we define $p_x := z_0$. If H_x contains some but not all of the vertices of C, then it contains just a subpath, since H_x is connected. For such vertices x we denote the first vertex of this path in clockwise orientation by p_x. If $H_x \cap C = \emptyset$, then there is a unique vertex p_x of H_x separating the rest of H_x from C.

By Theorem 6.1.2, it suffices to show that the catch digraph D of the family $((V(H_x), p_x)/x \in V(G))$ is an orientation of G. Let xy be an edge of G, that is, $H_x \cap H_y \neq \emptyset$. Let z be a vertex of $H_x \cap H_y$. If $H_x \cap C$ and $H_y \cap C$ are nonempty, then it is easy to see that $p_y \in V(H_x \cap C)$ or $p_x \in V(H_y \cap C)$. Thus $x \rightarrow y$ or $y \rightarrow x$ in D.

So suppose without loss of generality that $H_x \cap C = \emptyset$. Then there is exactly one path from z to C. Hence p_x lies on this path, and if $H_y \cap C = \emptyset$, then p_y does also. If $H_y \cap C = \emptyset$, then we may assume without loss of generality that p_x lies on the (p_y, z)-subpath. Now $p_x \in V(H_y)$ and $y \rightarrow x$ in D. If $H_y \cap C \neq \emptyset$, then the whole path from z to C must lie inside H_y, whence $y \rightarrow x$ in D. □

The converse is not true. The underlying graph of the in-tournament of Fig. 6.1 is not representable in any unicyclic graph. It can be easily shown that in any graph G representable in a unicyclic graph the following must hold: Any vertex x of an induced cycle of length at least 4 must be adjacent to at least one vertex from any other induced cycle in $G - x$. But this property is certainly not obeyed by the underlying graph ot the digraph of Fig. 6.1.

We believe that any graph orientable as an in-tournament is representable in a cactus – a connected graph in which any block is a cycle or an edge. Note that the opposite is not true: no cactus with at least two induced cycles of length at least four can be oriented as an in-tournament (every cactu can be represented in some subdivision of itself).

Theorem 6.3.2 has several consequences. We list some of them below.

Corollary 6.3.3 *Every chordal graph and every circular arc graph is orientable as an in-tournament.*

Proof: Chordal graphs are representable in trees (see [33]) and hence in unicyclic graphs. By definition, circular arc graphs are representable in a unicyclic graphs.

Now the claim follows from Theorem 6.3.2. □

A *Helly-representation* is a representation which has the so-called Helly-property — the total intersection of any family of pairwise intersecting representatives is nonempty.

Corollary 6.3.4 *Every graph with exactly one induced cycle of length greater than 3 is orientable as an in-tournament.*

Proof: By Theorem 6.3.2 it suffices to show that such a graph is representable in a unicyclic graph. Let G be a graph with only one nontrivial induced cycle $C = c_0c_1 \ldots c_{\ell-1}c_0$, $\ell \geq 4$. Let W be the set of vertices that are adjacent to all vertices of C and $T = V(G) - V(C) - W$. Since G contains exactly one induced cycle of length at least four, W induces a complete subgraph and vertices in T are adjacent to at most two consecutive vertices of C. Similarly, no two vertices of T with noncomparable neighbourhoods in C can be adjacent.

It is clear that there is a Helly representation of $G[V(C) \cup W]$ in a cycle of length ℓ. Also it is true that any Helly representation of $G - x$ in some unicyclic graph can be extended to another Helly representation of G in another unicyclic graph provided x is a simplicial vertex of G. So now it suffices to show that if $T \neq \emptyset$, then T contains a simplicial vertex.

First we prove that if T contains a vertex x which is not adjacent to any vertex of C, then T contains a simplicial vertex with this property. In fact, let S be a minimal $x - C$ separating set with A and B being the components of $G - S$ containing x and C respectively. Since S is minimal, each $y \in S$ is adjacent to some vertex in A and some vertex in B. Thus for any pair $u, v \in S$ there exists a path $ua_1 \ldots a_rv$ and a path $vb_1 \ldots b_tu$, where all $a_i \in A$ and all $b_j \in B$, such that these paths are chosen to be of smallest possible lengths. It follows that $ua_1 \ldots a_rvb_1 \ldots b_tu$ is a cycle of length at least four, which is distinct from C, implying that it must have a chord. But $a_ib_j \notin E(G)$ by definition of a vertex separating set, and $a_ia_j \notin E(G)$ and $b_ib_j \notin E(G)$ by the minimality of r and t. Thus the only possible edge is $uv \in E(G)$. Hence S is complete. Since $G[A \cup S]$ is chordal, by Dirac's Theorem (see [33]), $A \cup S$

contains two non-adjacent simplicial vertices or $G[A \cup S]$ is complete. Hence A must contain at least one simplicial vertex since $G[S]$ is complete.

Now assume that all vertices in T are adjacent to either one vertex or two consecutive vertices of C. If $x \in T$ is adjacent to c_i and c_{i+1} but not c_{i+2}, then $W \cup \{c_i, c_{i+1}\}$ is a $x - c_{i+2}$ separating set. Let A be the component in $G - (W \cup \{c_i, c_{i+1}\})$ containing x. Then $G[A \cup W \cup \{c_i, c_{i+1}\}]$ is chordal. Again, by Dirac's Theorem, it contains two non-adjacent vertices if it is not complete. So T must contain at least one simplicial vertex. If $x \in T$ is adjacent to c_i only, then either $W \cup \{c_i, c_{i+1}\}$ or $W \cup \{c_i, c_{i-1}\}$ is a $x - c_{i+2}$ separating set. By a similar discussion we can see that T contains at least one simplicial vertex. □

6.4 Strong In-tournament Orientability

Skrien [71] completely solved the problem concerning acyclic orientations of graphs without an induced subgraph from the set F for any $F \subseteq \mathcal{B}$, where $\mathcal{B}$ is the set of digraphs in Fig. 6.2. We now turn to the problem of orienting graphs as strong in-tournaments. Deciding whether a graph can be so oriented seems to be quite difficult. This is partly due to the fact that handling the strong connectivity requirement is not easy; for example, the class of graphs orientable as strong in-tournaments is not closed under induced subdigraphs. However, as we shall see below, for some classes of graphs, being orientable as a strong in-tournament is equivalent to being orientable as an in-tournament.

Proposition 6.4.1 *A graph without a separating complete subgraph (sometimes called a prime graph) is orientable as a strong in-tournament if and only if it is orientable as an in-tournament.*

Proof: One direction is trivial. For the other, let G be a graph without a separating complete subgraph, and let D be an orientation of G as an in-tournament with the minimum number k of strong components. We may assume $k \geq 2$. Let D_1 be the source component, and let D_2 be another strong component such that D_2 has exactly one predecessor in the strong component digraph $SC(D)$, namely D_1. Such a D_2 can be found by Theorem 6.2.6(b). Let V_1 be those vertices of D_1 that dominate the vertices of D_2. Again by Theorem 6.2.6, V_1 induces a tournament in D. If $V_1 = V(D_1)$, we are done since in that case we can reorient an arbitrary arc between D_1 and D_2 to obtain an in-tournament with fewer strong components, a contradiction. So let us assume $V_1 \neq V(D_1)$. By Theorem 6.2.6, there is no path between $V(D_1) - V_1$ and $V(D_2)$ which avoids V_1. Then V_1 induces a separating complete subgraph in G contradicting our assumptions. □

Note that there exist hamiltonian chordal graphs (and thus graphs orientable as in-tournaments (cf. Corollary 6.3.3)) which are not orientable as strong in-tournaments. Such an example is given in Fig. 6.3. It is clear that this example can be generalized to an infinite family. Although we are not able to solve the problem of characterizing those chordal graphs which are orientable as strong in-tournaments, we will mention a partial result.

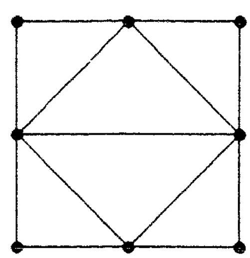

Figure 6.3: A hamiltonian chordal graph which is not orientable as a strong in-tournament

The following is a Corollary of the work in [49].

Proposition 6.4.2 *A graph G can be oriented as a strong local tournament if it is a proper circular arc graph which is not an interval graph.* □

Corollary 6.4.3 *A chordal graph is orientable as a strong local tournament if it is claw-free and net-free and not an interval graph.*

Proof: It was shown in [7] that a chordal graph is a proper circular arc graph if and only if it is claw-free and net-free. Now the claim follows from Proposition 6.4.2. □

Chapter 7

Oriented Graphs of Moon Type

An oriented graph D is *of Moon type* if every connected induced subgraph is either strong or acyclic. If D is also a tournament, then it is called a *tournament of Moon type* [35]. In [57], Moon gave a structural characterization of tournaments of Moon type. He proved that every tournament of Moon type can be obtained from a highly regular (cf. below) tournament by substituting transitive tournaments for the vertices. Tournaments of Moon type have also been studied by Burzio, Demaria, and Guido, [18, 35].

In this chapter, we give a similar structural characterization of oriented graphs of Moon type. Our characterization generalizes Moon's result. Specifically, we prove that every oriented graph of Moon type can be obtained from a local transitive tournament by substituting acyclic oriented graphs for the vertices.

In Section 7.1, we will mainly review previous results and some equivalent definitions of tournaments of Moon type. In Section 7.2, we shall discuss oriented graphs of Moon type and analyze several properties of such graphs. We also give some equivalent definitions of oriented graphs of Moon type, one of which implies a polynomial algorithm for recognizing these oriented graphs. Finally in Section 7.3, we prove our main result, which generalizes a theorem of Moon.

Let S be a subgraph of D and let $x \in D - S$. The vertex x *cones* S or S is *coned* by x if $x \rightarrow S$ or $S \rightarrow x$ whenever x is adjacent to a vertex of S. The subgraph S is *shrinkable* if S is coned by all vertices in $D - S$. A shrinkable subgraph S is *maximal* if it is not D and it is not properly contained in any shrinkable subgraph other than D. If S is shrinkable, then the vertices of S are said to be *equivalent*.

Suppose that the vertices of D are partitioned into vertex-disjoint subgraphs $S_1, S_2, \ldots, S_k$ of equivalent vertices. Then $S_i \rightarrow S_j$ or $S_j \rightarrow S_i$ if there is at least one arc between S_i and S_j. If D_k is an oriented graph on k vertices in which $v_i \rightarrow v_j$ if and only if $S_i \rightarrow S_j$, then we write $D = D_k(S_1, S_2, \ldots, S_k)$. An oriented graph is *simple* if there are no proper non-trivial subgraphs of equivalent vertices, that is, if the equation $D = D_k(S_1, S_2, \ldots, S_k)$ implies that $k = 1$ and $S_1 = D$, or $k = n$, $D = D_k$ and $S_i = v_i$.

For each subgraph B of D, the set of vertices which are dominated by at least one vertex of B is called the *outset* of B, denoted by $O(B)$; similarly the set of vertices which dominate at least one vertex of B is called the *inset* of B, denoted by $I(B)$.

7.1 Tournaments of Moon Type

In [35], a *tournament of Moon type* is defined to be a tournament in which each subtournament is hamiltonian or transitive. Note that a tournament is hamiltonian if and only if it is strong, and transitive if and only if it is acyclic. Thus our definition of an oriented graph of Moon type is consistent with this definition of a tournament of Moon type.

A tournament T is *highly regular* if the vertices T can be labeled as $v_1, v_2, \ldots, v_{2k+1}$ in such a way that $v_i \rightarrow v_j$ for all subscripts $i = 1, 2, \ldots, 2k+1$ and for all subscripts $j = i+1, i+2, \ldots, i+k \pmod{2k+1}$. It is easy to see that a highly regular tournament is a local transitive tournament.

The following theorem is reformulated from [18, 57].

Theorem 7.1.1 *The following statements are equivalent for a tournament T:*

(a) T is a tournament of Moon type;

(b) every subtournament of T is a tournament of Moon type;

(c) T is a local transitive tournament;

(d) $T = D_{2k+1}(S_1, S_2, \ldots, S_{2k+1})$, where D_{2k+1} is a highly reqular tournament and each S_i is a transitive tournament. □

If a local tournament is of Moon type, then we call it a *local tournament of Moon type*. From Theorem 7.1.1, we know that every tournament of Moon type is a local transitive tournament. The following proposition assures that every local tournament of Moon type is also a local transitive tournament.

Proposition 7.1.2 *An oriented graph is a local tournament of Moon type if and only if it is a local transitive tournament.*

Proof: Suppose that a local tournament D is not a local transitive tournament. Then by Lemma 3.2.5, D contains a forbidden quadruplet. Since a forbidden quadruplet is connected but neither strong nor acyclic, D is not of Moon type.

Suppose that D is a local transitive tournament. Then D is a local tournament. If D is not of Moon type, then D contains a connected subgraph S which is neither strong nor acyclic. Since S is not acyclic, S must contain at least one cycle. Let

$$C = v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_l \rightarrow v_1$$

be a longest cycle in S. Since S is connected and not strong, there exists a vertex $y \in V(S) - V(C)$ which is adjacent to a vertex, say v_i, in C. Suppose that y dominates v_i. (A similar discussion applies when $v_i \rightarrow y$.) Note that both y and v_{i-1} dominate v_i. The vertex y and the vertex v_{i-1} must be adjacent as D is a local tournament. Observe that v_{i-1} can not dominate y, as otherwise there is a cycle

$$v_1 \rightarrow \ldots \rightarrow v_{i-1} \rightarrow y \rightarrow v_i \rightarrow \ldots \rightarrow v_l \rightarrow v_1$$

of length $l + 1$, contradicting the choice of C. Thus y dominates v_{i-1}. Continuing this argument, we conclude that y dominates all vertices of C. But this is impossible as D is a local transitive tournament in which the outset of every vertex contains no cycle. Therefore D is of Moon type. □

Corollary 7.1.3 *Let T be a tournament. Then T is a local transitive tournament if and only if it is of Moon type.*

Proof: This follows immediately from Proposition 7.1.2. □

7.2 Oriented Graphs of Moon Type

The definition of an oriented graph of Moon type guarantees that every acyclic oriented graph is of Moon type. Nevertheless a strong oriented graph is not necessarily of Moon type. For example, an oriented graph formed by identifying two vertices from two distinct directed cycles is strong but not of Moon type.

Suppose that S is a subgraph of an oriented graph D which is of Moon type. Since every connected subgraph H of S is also a connected subgraph of D, H must be either strong or acyclic. Hence S is of Moon type. Conversely, if every subgraph of D is of Moon type, then D is of Moon type. Therefore we have the following proposition.

Proposition 7.2.1 *An oriented graph D is of Moon type if and only if every subgraph of D is of Moon type.* □

Proposition 7.2.2 *An oriented graph D is of Moon type if and only if $O(B) = I(B)$ for every strong subgraph B of D with $|V(B)| > 1$.*

Proof: Suppose that D is an oriented graph of Moon type and suppose that B is a strong subgraph of D with $|V(B)| > 1$. For each vertex x which is dominated by at least one vertex of B, x must dominate some vertex in B as otherwise $B + x$ would be a connected subgraph of D which is neither strong nor acyclic. Then $O(B) \subseteq I(B)$.

Similarly for each vertex x which dominates at least one vertex of B, x must be dominated by a vertex of B. Then $I(B) \subseteq O(B)$. Hence $O(B) = I(B)$.

Suppose that D is not of Moon type. Then there is a connected subgraph S which is neither strong nor acyclic. Let S' be a strong component of S of maximum size. Since S is not acyclic, we have $|V(S')| > 1$. Since S is not strong, we have $S' \neq S$. Note that S is connected. Then there exists a vertex $x \in S - S'$ such that x is adjacent to at least one vertex of S'. From the maximality of S', we have either $x \in O(S') - I(S')$ or $x \in I(S') - O(S')$. □

The following theorem turns out to be very useful in later discussions.

Theorem 7.2.3 *A connected oriented graph is of Moon type if and only if every (not necessarily connected) subgraph is either strong or acyclic.*

Proof: The sufficiency is obvious. To prove the necessity, suppose that D is of Moon type and S is a disconnected subgraph of D. We claim that each connected component of S is acyclic and hence S is acyclic.

Let $S_1, S_2, \ldots, S_k$ where $k > 1$ be the components of S. Without loss of generality, assume that S_1 contains a cycle. Since S_1 is a connected subgraph of D which is of Moon type, S_1 is strong. By hypothesis, the underlying graph G of D is connected. Thus there exists a path (in G) from S_1 to S_2. Let $x_1 \sim x_2 \sim \ldots \sim x_l$ be a shortest path from S_1 to S_2 in G. From the connectivity of S we conclude that $l > 2$. Since S_1 is strong, $S_1 + \{x_1, x_2, \ldots, x_l\}$ must be also strong. On the other hand, the only vertex in $S_1 + \{x_1, x_2, \ldots, x_l\}$ which is adjacent to x_l is x_{l-1}. Hence the vertex x_l has degree one in $S_1 + \{x_1, x_2, \ldots, x_l\}$. So $S_1 + \{x_1, x_2, \ldots, x_l\}$ can not be strong, a contradiction. Therefore S is acyclic. □

Proposition 7.2.4 *If D is a strong oriented graph of Moon type, then every longest directed path induces a strong subgraph.*

Proof: Suppose $L = x_1 \to x_2 \to \ldots \to x_l$ is a longest path in D. Since D is strong, x_l must dominate some vertex in D. Since L is a longest path, x_l can only dominate vertices in L. Thus L contains a cycle and hence it must strong. □

We have seen from Theorem 7.1.1 that all tournaments of Moon type are local transitive tournaments, that is, the outset as well the inset of each vertex is a transitive tournament. For general oriented graphs of Moon type, there is a nice local property for each vertex.

Proposition 7.2.5 *If D is an oriented graph of Moon type, then the outset as well the inset of every vertex is acyclic.*

Proof: If the outset (or the inset) of some vertex x contains a cycle, then this cycle together with x induces a connected subgraph which is neither strong nor acyclic. □

The following theorem will imply a polynomial algorithm to recognize all oriented graphs of Moon type. Define $O^*(x) = V(D) - I[x]$ and $I^*(x) = V(D) - O[x]$. We call $O^*(x)$ the *super-outset* of x and $I^*(x)$ the *super-inset* of x.

Theorem 7.2.6 *A connected oriented graph is of Moon type if and only if the super-outset as well as the super-inset of each vertex is acyclic.*

Proof: Suppose that D is of Moon type. No vertex in $O^*(x)$ dominates x, so $O^*(x) \cup \{x\}$ can not be strong and hence, by Theorem 7.2.3, $O^*(x) \cup \{x\}$ must be acyclic, and therefore also $O^*(x)$. Similarly $I^*(x)$ is acyclic. Conversely, suppose that D is not of Moon type. By Proposition 7.2.2, there exists a strong subgraph B ($|B| > 1$) for which $O(B) \neq I(B)$. If $O(B) - I(B) \neq \emptyset$, letting $x \in O(B) - I(B)$, then $I^*(x)$ is not acyclic as B contains at least one cycle. A similar argument applies if $I(B) - O(B) \neq \emptyset$. □

Corollary 7.2.7 *There exists a polynomial algorithm to recognize oriented graphs of Moon type.*

Proof: There exists a linear time algorithm (cf. [1]) to test whether an oriented graph is acyclic or not. Thus to test whether $I^*(x)$ and $O^*(x)$ contain a cycle, for any vertex x, takes $O(mn)$ time, where n and m denote the numbers of vertices and arcs respectively. □

It is well known that every strong tournament T on at least 4 vertices has a vertex x such that $T - x$ is still strong (cf. [58]). For a local tournament D, if D is strong and not a directed cycle, there exists a vertex x such that $D - x$ is still strong (cf. [4]). The following more general theorem of this type is an easy consequence of Theorem 7.2.3.

Theorem 7.2.8 *A connected oriented graph D is of Moon type if and only if for every ordering of vertices of D, $v_1, v_2, \ldots, v_n$, the following property holds: for some $0 \leq k \leq n$, $V - \{v_1, v_2, \ldots, v_i\}$ is strong for $i \leq k$ and is acyclic for $i > k$.* □

7.3 Oriented Graphs of Moon Type and Local Tournaments

We have seen that a tournament $T = D_k(S_1, S_2, \ldots, S_k)$, $1 < k < n = |T|$, is of Moon type if and only if D_k is of Moon type and each S_i is transitive. A similar statement holds for general oriented graphs. Let $n = |D|$.

Proposition 7.3.1 *Let $D = D_k(S_1, S_2, \ldots, S_k)$, $1 < k < n$, be connected. Then D is of Moon type if and only if D_k is of Moon type and each S_i is acyclic.*

Proof: Suppose that D is of Moon type. If D_k is not of Moon type, then there exists a connected subgraph S in D_k which is neither strong nor acyclic. For each vertex v_i of S, arbitrarily choose a vertex from S_i corresponding to v_i. Then the subgraph of D induced by these vertices is connected but neither strong nor acyclic, contradicting the assumption. Therefore D_k is of Moon type. Now suppose some S_i contains a cycle. Since D is connected and $k > 1$, there exists a vertex $x \notin S_i$ which is adjacent to some vertex in S_i. Then we must have either $x \to S_i$ or $S_i \to x$ in D. But then by Proposition 7.2.5 D is not of Moon type, contradicting the hypothesis.

Conversely, suppose that D_k is of Moon type and each S_i is acyclic. Let x be any vertex in D. Then x is in some S_i. Let v_i be the vertex of D_k corresponding to S_i. Since D_k is of Moon type, $O^*(v_i)$ is acyclic by Theorem 7.2.6. In fact, $O^*[v_i]$ is acyclic. Suppose that S' is the subgraph of D induced by $\bigcup_{v_j \in O^*[v_i]} V(S_j)$. Then S' must be acyclic in D because each S_j is acyclic. It is easy to see that $O^*(x)$ in D is a subgraph of S'. So $O^*(x)$ is also acyclic. Similarly $I^*(x)$ is acyclic and hence, by Theorem 7.2.6, D is of Moon type. □

Proposition 7.3.2 *Let D be a connected strong oriented graph. Then no two distinct maximal shrinkable subgraphs contain a common vertex.*

Proof: Let S_1 and S_2 be any two distinct maximal shrinkable subgraphs in D with $x \in S_1 \cap S_2$. We claim first that $V(S_1) \cup V(S_2) \neq V(D)$. Assume to the contrary that $V(S_1) \cup V(S_2) = V(D)$. Since D is connected, there is a vertex y in $S_2 - S_1$ which is adjacent to at least one vertex in S_1. Then either $y \to S_1$ or $S_1 \to y$ because S_1 is shrinkable. Assume $y \to S_1$. (A similar argument applies when $S_1 \to y$.) If there is a vertex $z \in S_1 - S_2$ and a vertex $w \in S_2$ such that $z \to w$, then $z \to S_2$ as S_2 is shrinkable. In particular, $z \to y$, contradicting the fact that $y \to S_1$. Hence no vertex in $S_1 - S_2$ dominates a vertex in S_2 and $S_1 \cup S_2 = D$ is not strong, a contradiction to the hypothesis.

To complete the proof, suppose that $y \in D - (S_1 \cup S_2)$ is a vertex which is adjacent to at least one vertex in $S_1 \cup S_2$, say to a vertex in S_1. Then $y \to S_1$ or $S_1 \to y$, in particular, $y \to x$ or $x \to y$. Hence $y \to S_2$ or $S_2 \to y$. Therefore $S_1 \cup S_2$ is a shrinkable subgraph which strictly contains S_1, contradicting the maximality of S_1. Therefore S_1 and S_2 have no common vertex. □

Let D be an oriented graph of Moon type. From the above proposition, we know that for each vertex x of D there exists a unique maximal acyclic shrinkable subgraph S_x containing x, such that some k of these subgraphs, say $S_1, S_2, \ldots, S_k$, form a partition of D. (Equivalent vertices x and y will have $S_x = S_y$.) Hence each oriented graph D can be written, in an essentially unique way, as $D = D_k(S_1, S_2, \ldots, S_k)$. We call $D_k(S_1, S_2, \ldots, S_k)$ the *canonical expression*. It is easy to see that D_k is simple.

Lemma 7.3.3 *Let D be a connected oriented graph of Moon type and let C be a directed cycle in D. Then each vertex of D must have at least one in-neighbour and at least one out-neighbour in C.*

Proof: Since D is a connected oriented graph of Moon type, by Theorem 7.2.3, $C \cup \{x\}$ is strong as it is not acyclic. Hence there is at least one vertex in C dominating x and at least one vertex in C being dominated by x. □

The following theorem conjectured by Hell [38] is the main result of this chapter.

Theorem 7.3.4 *Let D be an oriented graph with the canonical expression $D_k(S_1, S_2, \ldots, S_k)$. Then D is of Moon type if and only if D_k is a local transitive tournament and each S_i is acyclic.*

Proof: For the sufficiency, suppose that D_k is a local transitive tournament and each S_i is acyclic. Then D_k is of Moon type by Proposition 7.1.2, and hence D is of Moon type by Proposition 7.3.1.

For the necessity, suppose that D is of Moon type. Without loss of generality, assume that D is connected otherwise we consider each component of D. Then D_k is connected. If D is acyclic, then $k = 1$, and so D_k has only one vertex and it is trivially a local transitive tournament. If D is strong, then D_k must be strong. By Proposition 7.3.1 each S_i is acyclic, and by Propositions 7.3.1 and 7.1.2 it suffices to show that D_k is a local tournament. Suppose to the contrary that D_k is not a local tournament. Then in D_k there exists a vertex which has two non-adjacent out-neighbours or there exists a vertex which has two non-adjacent in-neighbours. Since the two cases are symmetric, assume that there is a vertex z which has non-adjacent out-neighbours x and y. We claim that there exists an acyclic shrinkable subgraph containing vertices x and y.

Since D_k is strong, there exist directed paths from x to z and from y to z. Among all directed paths from x to z and from y to z, choose a shortest one. Without loss of generality, let

$$P: \ x \to x_1 \to x_2 \to \ldots \to x_k = z$$

be such a path. Note that $x \to x_1 \to \ldots \to x_k \to x$ is a directed cycle. By Lemma 7.3.3, $y \to x_i$ for some $i = 1, 2, \ldots, k$. If $i \neq 1$, then $y \to x_i \to x_{i+1} \to \ldots x_k = z$ is a path from y to z of length $< k$, contradicting the choice of P. Thus $i = 1$.

Among the vertices $x_2, x_3, \ldots, x_k$, let x_l be the one of the smallest subscript such that $x_l \to x$ or $x_l \to y$. If $x_l \to x$, then $x \to x_1 \to \ldots x_l \to x$ is a cycle. By Lemma 7.3.3, $x_j \to y$ for some j with $2 \leq j \leq l$. Since l is the smallest subscript, $j = l$. Similarly if $x_l \to y$ then $x_l \to x$. Thus, in D_k, $x_l \to \{x, y\} \to x_1$ and no vertex x_i with $1 < i < l$ is adjacent to x or y. Moreover $x_1 \to x_2 \to \ldots \to x_l$ is a directed path.

Let $S = \{v \in V(D_k) |\ x_l \to v \to x_1$, and v is not adjacent to x_i for any i with $1 < i < l\}$. Then $\{x, y\} \subseteq S$. Let $S' \subseteq S$ be a subset of the smallest cardinality which contains both x and y and is coned by all vertices in $S - S'$. Now $S' \in O(x_l)$ and hence S' induces an acyclic subgraph in D_k by Proposition 7.2.5. We claim that S' is shrinkable in D_k.

Suppose that $w \notin S'$ is a vertex dominated by some vertex $v \in S'$. We will show that w is dominated by all vertices of S'. One can show, applying a similar argument, that if $w \notin S'$ dominates some vertex in S' then w dominates all vertex of S'.

Without loss of generality, we assume that $w \notin S$ (since S' is shrinkable in S). By Lemma 7.3.3, $w \to x_i$ for some $1 \leq i \leq l$ as $v \to x_1 \to \ldots \to x_l \to v$ is a cycle. Suppose that w is not dominated by some vertex $u \in S'$. Consider the cycle $C' = u \to x_1 \to \ldots \to x_l \to u$. Since w dominates x_i of C', w must be dominated by a vertex x_j of C' by Lemma 7.3.3.

Now let i and j be chosen so that $x_i \in \{x_1, x_2, \ldots, x_l\}$ is the vertex with the greatest index such that $w \to x_i$ and $x_j \in \{x_1, x_2, \ldots, x_l\}$ is the vertex of the smallest subscript such that $x_j \to w$. Since $w \notin S$, it is not the case that $i = 1$ and $j = l$. Hence we have the following two cases.

Case 1. Suppose that $i = 1$ and $j < l$. Then $w \to x_1 \to \ldots \to x_j \to w$ is a directed cycle and $v \to \{w, x_1\}$ is not adjacent to x_k with $1 < k \leq l$. Hence $\{v, w, x_1, \ldots, x_j\}$

induces a connected subgraph which is neither strong nor acyclic, contradicting the fact that D_k is of Moon type.

Case 2. Suppose that $i > 1$. Then, for each $a \in S'$ such that $a \rightarrow w$, $a \rightarrow w \rightarrow x_i \rightarrow \ldots \rightarrow x_l \rightarrow a$ is a cycle, and, for each $b \in S'$, b must dominate some vertex in this cycle by Lemma 7.3.3. If b does not dominate w, then $b \rightarrow a$ as b does not dominate any x_k $(i \leq k \leq l)$ either.

Suppose that not all vertices of S' dominate w and let $S'_1 = \{c \in S' | \ c \rightarrow w\}$ and $S'_2 = \{c \in S' | \ c \not\rightarrow w\}$. Then $S'_1 \cup S'_2 = S'$ and from the above discussion we have $S'_1 \neq \emptyset$, $S'_2 \neq \emptyset$, and $S'_1 \rightarrow S'_2$. Since x and y are not adjacent, exactly one of S'_1 and S'_2 contains both x and y. Without loss of generality, let S'_1 contain both x and y. Then $S'_1 \subset S$ is coned by all vertices in $S - S'_1$ with $|S'_1| < |S'|$, which contradicts the choice of S'.

Therefore S' induces an acyclic shrinkable subgraph in D_k and $1 < |S'| < |D_k|$. This contradicts the fact that D_k is simple. So D_k is a local tournament, and this completes the proof of the theorem in view of the earlier observations. □

From Theorem 7.3.4, we know that all oriented graphs of Moon type can be generated from a local transitive tournament by substituting acyclic oriented graphs for the vertices.

Bibliography

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, **The Design and Analysis of Computer Algorithms**, Addison-Wesley, Reading, Massachusetts (1974).

[2] B. Alspach and C. Tabib, A Note on the number of 4-circuits in a tournaments, in **Annals of Discrete Mathematics** 12 (1982) 13 - 19.

[3] A. Apostolico and S.E. Hambrusch, Finding maximum cliques on circular-arc graphs, Inform. Proce. Letters 26 (1987) 209 - 215.

[4] J. Bang-Jensen, Locally semicomplete digraphs–a generalization of tournaments, J. Graph Theory, Vol. 14, No. 3 (1990) 371 - 390.

[5] J. Bang-Jensen, Edge-disjoint in- and out-branchings in tournaments and related path problems, J. Combinatorial Theory (B) 51, (1991) 1 - 23.

[6] J. Bang-Jensen, On digraphs with pathmerge property, submitted.

[7] J. Bang-Jensen and P. Hell, Chordal proper circular arc graphs, (1991), submitted.

[8] J. Bang-Jensen and P. Hell, Hamiltonian algorithms for in-tournaments, (1991), submitted.

[9] J. Bang-Jensen, P. Hell, and J. Huang, Local tournaments and proper circular arc graphs, CSS/LCCR TR90 - 11, Simon Fraser University, Revised Aug. 1990.

[10] J. Bang-Jensen, J. Huang, and E. Prisner, In-tournaments, TR 91 - 030, Universität Bielefeld, Germany, (1991).

[11] J. Bang-Jensen, J. Huang, and E. Prisner, In-tournament digraphs, J. Combinatorial Theory (B) (to appear).

[12] D. Bauer and R. Tindell, Graphs isomorphic to subgraphs of their line-graphs, Disc. Math. 41 (1982) 1 - 6.

[13] L. Beineke and K.B. Reid, **Tournaments, Selected Topics in Graph Theory** Academic Press, N. Y. (1979) pp. 169 - 204.

[14] C. Benzaken, Y. Crama, P. Duchet, P. L. Hammer, and F. Mafray, More characterizations of triangulated graphs, J. Graph Theory, Vol. 4 (1990) 413 - 412.

[15] S. Benzer, On the topology of the genetic fine structure, Proc. Nat. Acad. Sci. 45 (1959) pp. 1607 - 1620.

[16] J.A. Bondy and U.S.R. Murty, **Graph Theory and Applications**, American Elsevier, N. Y. (1976).

[17] K.S. Booth and G.S. Lueker, Testing for the consecutive ones property, interval graphs and graph planarity using PQ-tree algorithm, J. Computer and System Sciences 13, (1976) 335 - 379.

[18] M. Burzio and D.C. Demaria, Characterization of tournaments by coned 3-cycles, Acta Universitatis Carolinanae-Mathematica Et Physica, Vol. 28, No. 2 (1987).

[19] P. Damaschke, Forbidden Ordered Subgraphs, manuscript.

[20] X. Deng, P. Hell, and J. Huang, Recognition and representation of proper circular arc graphs, **Integer Programming and Combinatorial Optimization**, Proceeding of the second IPCO conference, (Egon Balas, G. Cornuéjols, and R. Kannan, eds.) (1992) pp. 114 - 121.

[21] G.A. Dirac, On rigid circuit graphs, Abh. Math. Sem. Univ. Hamburg 25, 71 - 76.

[22] P. Duchet, M. Las Vergnas, and H. Meyniel, Connected cutsets of a graph and triangle bases of the cycle space, Disc. Math. 62 (1982) 145 - 154.

[23] M. Farber. Characterizations of strongly chordal graphs. Disc. Math. 43 (1983) 173 - 189.

[24] D.R. Fulkerson and O.A. Gross, Incidence Matrices and interval graphs, Pacific J. Math. 15 (1965) 835 - 855.

[25] T. Gallai, Transitiv orientierbare graphen, Acta Math. Acad. Sci. Hungar, 18, (1967) 25 - 66.

[26] M.R. Garey and D.S. Johnson, **Computers and Intractability** W.H. Freeman and Company, N. Y. 1979.

[27] M.R. Garey, D.S. Johnson, G.L. Miller, and C.H. Papadimitriou, The complexity of coloring circular arcs and chords, Siam J. Algebraic and Disc. Math. 2 (1980) 216 - 277.

[28] F. Gavril, Algorithms on circular-arc graphs, Networks 4, 357 - 369

[29] F. Gavril, The intersection graphs of subtrees in trees are exactly the chordal graphs, J. Combinatorial Theory (B) 16 (1974) 47 - 56.

[30] F. Gavril, Algorithms for minimum coloring, maximum clique, minimum covering by cliques and maximum independence set of a chordal graph, Siam J. Computer Vol. 1, No.2, June 1972.

[31] A. Ghouilà-Houri, Caractérisation des graphes non orientès dont on peut orienter les arrêtes de maniere à obtenir le graphe d'une relation d'ordre. C. R. Acad. Sci. Paris 254 (1962) 1370 - 1371.

[32] P.C. Gilmore and A.J. Hoffman, A characterization of comparability graphs and of interval graphs, Canad. J. Math. 16 (1964) 539 - 548.

[33] M.C.Golumbic, **Algorithmic Graph Theory and Perfect Graphs**, Academic Press, N. Y. 1980.

[34] M.C. Golumbic and P.L. Hammer, Stability in circular arc graphs, J. Algorithm 9 (1988) 314 - 320.

[35] C. Guido, Structure and reconstructure of Moon tournaments, to appear.

[36] U.I. Gupta, D.T. Lee and J.Y.-T. Leung, Efficient algorithms for interval graphs and circular arc graphs, Networks 12 (1982) 459-467.

[37] F. Harary, **Graph Theory**, Addison-Wesley, Reading, MA (1969).

[38] P. Hell, Personal communication.

[39] P. Hell, J. Bang-Jensen, and J. Huang, Local tournaments and proper circular arc graphs, in **Algorithms, Vol. 450, Springer, Lecture Notes in Compter Science**, T. Asano, T. Ibaraki, H. Imai, and T. Nishizeki (eds.) pp. 101 - 109.

[40] P. Hell and J. Huang, On the lexicographic method for recognizing comparability graphs, proper circular arc graphs, and proper interval graphs, in preparation.

[41] P. Hell and J. Huang, Linear algorithms for c-colouring and for finding maximum cliques in proper circular arc graphs, submitted.

[42] W.L. Hsu, $O(mn)$ isomorphism algorithms for circular arc graphs, manuscript.

[43] W.L. Hsu, Maximum weight clique algorithms for circular arc graphs and circle graphs, Siam J. Comp. 14 (1985) 224 - 231.

[44] W.L. Hsu, A simple test of interval graphs, A lecture in the Sixth Siam Conference on Discrete Mathematics, 1992.

[45] W.L. Hsu, W.K. Shih, and T.C. Chern, An $O(n^2 \log n)$ algorithm for the hamiltonian cycle problem on circular arc graphs, to appear.

[46] W.L. Hsu and K.H. Tsai, Linear time algorithms on circular arc graphs, Information Processing Letters 40 (1991) 123 - 129.

[47] J. Huang, Structure theorems on local tournaments, in preparation.

[48] J. Huang, Oriented graphs of the Moon type, to be submitted.

[49] J. Huang, A result on local tournaments, unpublished manuscript (1989).

[50] R.M. Karp, Reducibility among combinatorial problems, **Complexity of Computer Computations**, R.E. Miller and J.W. Thatcher (eds.), Plenum Press, N.Y., pp. 85 - 103 (1972).

[51] V. Klee, What are the intersection graphs of arcs in a circle, Amer. Math. Monthly 76 (1969) 810 - 813.

[52] D. Knuth, **Art of Computer Programming**, Vol. 3, Sorting and Searching, Addison-Wesley (1973).

[53] N. Korte and R.H. Möhring, An incremental linear-time algorithm for recognizing interval graphs, Siam J. Comput., Vol. 18, No. 1, (1989) 68 - 81.

[54] H. Maehara, A digraph represented by a family of boxes or spheres, J. Graph Theory 8 (1984) 431 - 439.

[55] Y. Manoussakis, Indifference graphs, in **Proof techniques in Graph Theory** (F. Harary ed.), Academic Press (1969) pp. 139 - 146.

[56] S. Masuda and K. Nakajima, An optimal algorithm for finding a maximum independent set of a circular arc graph, Siam J. Comput. Vol. 17, No. 1, Feb. 1988.

[57] J.W. Moon, **Topics on Tournaments**, Holt, Reinhard and Winston, N. Y. (1969).

[58] J.W. Moon, Tournaments whose subtournaments are irreducible or transitive, Canad. Math. Bull. Vol. 21 (1), 1979.

[59] C. Morrow and S. Goodman, An efficient algorithm for finding a longest cycle in a tournament, **Proc. 7th Southeastern Conference on Combicatorics, Graph Theory, and Computing,** Utilitas Mathematics Publishing, Wennipeg, pp. 453 - 462 (1976).

[60] S. Olariu, An optimal greedy heuristic to color interval graphs, Inform. Proce. Letters 37 (1991) 21 - 25.

[61] J.B. Orlin, M.A. Bonuccelli, and D.P. Bovet, An $O(n^2)$ algorithm for coloring proper circular arc graphs. Siam J. Algorithm Disc. Math. Vol. 2 (1981) 88 - 93.

[62] C.H. Papadimitriou and K.Steiglitz, **Combinatorial Optimization, Algorithms and Complexity**, Englewood Cliffs, N. J., Prentice Hall (1982).

[63] A. Pnueli, A. Lempel, and S. Even, Transitive orientation of graphs and identification of permutation graphs, Canad. J. Math. 23 (1971) 160 - 175.

[64] E. Prisner, Familien zusammenhängender Teilgraphen eines Graphen und ihre Durchschnittsgraphen, Dissertation Hamburg, 1988.

[65] E. Prisner, Intersection-representation by connected subgraphs of some n-cyclomatic graph, to appear in *Ars Combinatoria*.

[66] E. Prisner, Homology of the line graph and of related graph-valued functions, to appear in *Archiv der Mathematik*.

[67] F.S. Roberts, **Discrete Mathematical Models**, Prentce-Hall, Englewood Cliff, NJ (1976).

[68] P.L. Roberts, Indifference graphs, in **Proof Techniques in Graph Theory** (F. Harary, ed.), pp. 139 - 146, Academic Press, N. Y. (1969).

[69] D.J. Rose, R.E. Tarjan, and G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, Siam J. Comput. Vol. 5, No. 2, June 1976.

[70] W.K. Shih and W.L. Hsu, An $O(n^{\frac{3}{2}})$ algorithm to color proper circular arcs, Disc. Appl. Math. 25 (1989) 321 - 323.

[71] D.J. Skrien, A relationship between triangulated graphs, comparability graphs, proper interval graphs, proper circular arc graphs, and nested interval graphs, J. Graph Theory, Vol. 6 (1982) 309 - 316.

[72] J. Spinrad, On comparability and permutation graphs, Siam J. Comput., Vol. 14, No. 3 (1985) 658 - 670.

[73] F.W. Stahl, Circular genetic maps, J. Cell Physiology, 70 (suppl. 1) (1967) 1 - 12.

[74] K.E. Stouffers, Scheduling of traffic lights- a new approach, Transportation Res., 2 (1968) 199 - 234.

[75] R.E. Tarjan, Finding optimum branchings, *Networks* 7, (1977), 25 - 25.

[76] R.E. Tarjan and M. Yannakakis, Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, Siam J. Computing 13 (1984) 566 - 579.

[77] A. Teng and A. Tucker, An $O(qn)$ algorithm to q- color a proper family of circular arcs, Disc. Math. 55 (1985) 233 - 243.

[78] C. Thomassen, Edge-disjoint hamiltonian paths and cycles in tournaments, Proc. London Math. Soc. (3) 45 (1982) pp. 151 - 168.

[79] A. Tucker, Coloring a family of circular arcs, Siam, J. Appl. Math., 29 (1975), pp. 493 - 502.

[80] A. Tucker, An effiencient test for circular arc graphs, Siam J. Comput. Vol. 9, No. 1, Feb. 1980.

[81] A. Tucker, Matrix characterizations of circular arc graphs, Pacific J. of Math. Vol.39, No. 2 (1971) 535 - 545.

[82] A. Tucker, Structure thorem for some circular arc graphs, Disc. Math. 7 (1974) 167 - 195.

[83] J. Urrutia and F. Gavril, An algorithm for fraternal orientation of graphs, Information Processing Letters 41 (1992) 271 - 274.

[84] J. Urrutia, Personal communication.