CHAPTER I
PRELIMINARIES

1.1. Introduction

The elementary concepts of graphs, sub graphs, digraphs, oriented and labeled graphs, isomorphisms, etc., are mainly taken from [Go 80], [Fo 92] and [CH 95].

The study of intersection graphs was started in the early 1950's. Chordal graphs are very important type of intersection graphs. A major investigation of chordal graphs under the framework of intersection graphs was done by Gavril [Ga 74], [Ga 75], [Ga 78]. It was Gavril [Ga 74] who proved the very important result that chordal graphs are intersection graphs of subtrees of a tree. Interval graphs and circular arc graphs are very important types of subtree graphs and were studied by Gilmore and Hoffman [GH 64], Lekkerkerker and Boland [LB 62], Fulkerson and Gross [FG 65], [KI 69], [Tuc 71].

The definitions and notations followed in this chapter are from [BM 76], [Go 80], [Ha 69], [CH 95], [An 94], and [Ta 99].

1.2 Basic Concepts

Let G (V, E) be a graph with vertex set \( V = \{v_1, v_2, ..., v_n\} \) and edge set \( E = \{e_1, e_2, ..., e_m\} \). A chordless cycle is a simple cycle \((v_0, v_1, ..., v_k, v_0)\) in \( G(V, E) \), where \((v_i, v_j) \notin E\) for \( i \) and \( j \) differing by more than 1.

A tree is a connected graph, with no cycles.

If \( C = v_0 e_1 v_1 e_2 ... e_{k-1} v_k v_0 \) is a cycle in \( G(V, E) \), a chord of \( C \) is an edge \( e \) in \( E(G) \) connecting vertices \( v_i \) and \( v_j \) such that \( e \neq e_i \) for \( i = 1, 2, 3, ..., k \).
A graph is **chordal** if every cycle containing at least four vertices has a chord. An edge when its removal splits the graph $G$ into different components is called a **cut edge** of $G$.

A connected graph is a tree if and only if, every edge is a cut edge. An **edge cut** of $G$ is a subset of $E$ of the form $[S, \overline{S}]$, where $S$ is a nonempty proper subset of $V$ and $\overline{S} = V \setminus S$, where $[S, \overline{S}]$ denotes the set of all edges with one end in $S$ and the other in $\overline{S}$. A minimal edge cut of $G$ is called a **bond**. A vertex $v$, of $G$ is a **cut vertex**, if the removal of $v$ splits the graph $G$ into different components. A **vertex-cut** of $G$ is a subset $V'$ of $V$ such that, $G - V'$ is disconnected. A **$k$-vertex cut** is a vertex cut of $k$ elements. A complete graph has no vertex cut.

### 1.3. Undirected and Directed Graphs

The **reversal** of a graph $G (V, E)$ is the graph $G^{-1} = (V, E^{-1})$, where $E^{-1} = \{ (x, y) | (y, x) \in E \}$. The **symmetric closure** of a graph $G$ is the graph $G (V, E)$, where $E = E \cup E^{-1}$. A graph $G (V, E)$ is called undirected if its adjacency relation is symmetric, i.e., $E = E^{-1}$. Equivalently, $E = E \cup E^{-1}$.

A graph $H(V, F)$ is called an oriented (directed) graph, if its adjacency relation is anti symmetric, i.e., $F \cap F^{-1} = \emptyset$.

A **directed graph** $D$ is an ordered triple $(V(D), A(D), \psi_D)$ where $V(D)$ is a nonempty set of vertices, $A(D)$, disjoint from $V(D)$, is a set of arcs, and an incidence function $\psi_D$ that associates each arc of $D$, with an ordered pair of (not necessarily distinct) vertices of $D$. An arc $a = (u, v)$ is incident
with u and v, and the vertex u is the tail of 'a' and v is its head; 'a' is an in-arc of v and an out-arc of u. A directed graph or digraph is strict if it has no loops and no two arcs with the same head and tail, has the same orientation.

If there is a directed uv - path in D, vertex v is said to be reachable from vertex u in D. Two vertices are disconnected in D, if each is reachable from the other. A digraph D is disconnected if it has exactly one dicomponent.

Two digraphs D and D' are isomorphic if there is a bijection \( \phi \) from V(D) to V(D'), such that the arc \((u, v)\) is in E(D) if and only if arc \((\phi(u), \phi(v))\) is in E(D'). An isomorphism from D to itself is an automorphism.

1.4. Directed walk, Path, Cycle:

A directed walk is a sequence \( W = v_0 a_1 a_2 ... a_k v_k \) of vertices and arcs such that \( a_i = (v_{i-1}, v_i) \). A directed trail is a directed walk that is a trail, whose edges are disjoint. A directed path is a directed walk in which all vertices are distinct. A closed directed trail is a directed tour. A directed path with first vertex u and final vertex v is a uv - directed path. The length of a directed path is the number of arcs in it.

With each digraph D, we can associate a graph G on the same vertex set; corresponding to each arc o : D there is an edge of G with the same ends. This graph is the underlying graph of D. Conversely, for any given graph G, we can obtain a digraph from C, by specifying for each link, an order on its ends. Such a digraph is called an orientation of G. An orientation of a complete graph is called a tournament.
1.4.1. Hamiltonian Path

A directed Hamilton path of \( D \) is a directed path that includes every vertex of \( D \). It is known that, every tournament has a directed Hamilton path. A directed cycle is a directed tour of length \( k > 1 \) in which \( v_0 = v_k \) and vertices \( v_0, v_1, \ldots, v_{k-1} \) are distinct. A directed Hamilton cycle of \( D \) is a directed cycle that includes every vertex of \( D \).

A directed acyclic graph is a digraph \( D \) with no cycles. Let \( u, v \) be the vertices in \( V(D) \). The vertex \( u \) is an ancestor of \( v \); and \( v \) is a descendant of \( u \), if there is a \( uv \) – directed path in \( D \); otherwise \( u \) and \( v \) are unrelated. If \( u \) is an ancestor of \( v \) and \((uv)\) is an arc, then \( u \) is a parent of \( v \), and \( v \) is a child of \( u \).

1.5. Rooted Tree

A rooted tree is a directed acyclic graph in which all vertices have indegree one, and a specially designated node, called the root, has indegree zero. The root of a rooted tree \( T \) is denoted by \( \text{root}(T) \). The subtree of \( T \) rooted at \( v \) is the subtree of \( T \) induced by the descendants of \( v \).

The in-degree \( d_-(v) \) of a vertex \( v \) in \( D \) is the number of arcs with head \( v \); and out-degree \( d_+(v) \) of \( v \) is the number of arcs with tail \( v \). We denote the minimum and maximum in-degrees and out-degrees of \( D \) by \( \delta^-(D) \), \( \Delta^-(D) \), \( \delta^+(D) \) and \( \Delta^+(D) \) respectively. Also, a directed graph \( G(V,E) \) is rooted at vertex \( r \) if there is a path from \( r \) to every vertex in \( V \). \( G(V,E) \) is also called a rooted directed acyclic graph with root \( r \).
1.6. Clique

A subset $A \subseteq V$ of $r$ vertices is an $r$-clique if it induces a complete subgraph. That is, if $A \subseteq V$ is an $r$-clique, then $G[A] \cong K_r$. A single vertex is a 1-clique. A clique $A$ is maximal, if there is no clique of $G$ which properly contains $A$. A clique is maximum, if there is no clique of $G$ of larger cardinality. The clique number $\omega(G)$ is the number of vertices in a maximum clique of $G$. A clique cover of size $k$ is a partition of the vertices, $V = A_1 + A_2 + \ldots + A_k$, such that each $A_i$ is a clique. The clique cover number of $G$, $k(G)$ is the size of a smallest possible clique cover of $G$.

1.7. Independent set or Stable set

An independent set or a stable set of a graph $G$ is a subset $X$ of vertices, where no two of which are adjacent. The stability number of $G$, $\alpha(G)$, is the number of vertices in a stable set of maximum cardinality.

A proper $c$-coloring is a partition of the vertices $V = X_1 + X_2 + \ldots + X_c$, such that each $X_i$ is a stable set. Here the members of $X_i$ are painted with the color $i$, and adjacent vertices will receive different colors.

The chromatic number of $G$, $\chi(G)$, is the smallest possible number $c$, for which there exists a proper $c$-coloring of $G$.

Corresponding to the above properties of a graph $G$, we have the following results; $\omega(G) \leq \chi(G)$ and $\alpha(G) \leq k(G)$. Since every vertex of a maximum clique (maximum stable set) must be contained in a different partition segment in any minimum proper coloring (minimum clique cover), we get, $\omega(G) = \alpha(G)$ and
\( \chi(G) = \kappa(G) \). A set \( S \) is an independent set if \( G[S] \) is a null graph. For any graph, the intersection of a clique and a stable set can be at most one vertex. Also, for any graph, \( G, \omega(G) \leq \chi(G) \) and \( \alpha(G) \leq \kappa(G) \). These inequalities are dual to one another since, \( \alpha(G^c) = \omega(G) \) and \( \kappa(G) = \chi(G^c) \). Cliques and Independent sets have been studied by Fred Galvin [Gal 2000].

1.8. Perfect graph

Let \( G (V, E) \) be an undirected graph. The graph which satisfies the properties, (i) \( \omega(G[A]) = \chi(G[A]) \) for all \( A \subseteq V \); (ii) \( \alpha(G[A]) = \kappa(G[A]) \) for all \( A \subseteq V \), is called perfect.

Also, a graph is called \( \chi \) - perfect, if it satisfies (i), and \( \alpha \) - perfect if it satisfies (ii).

The Perfect Graph theorem states that a graph is \( \chi \) - perfect if and only if it is \( \alpha \) - perfect. It is clear by duality that a graph \( G \) satisfies (i) if and only if its complement \( G^c \) satisfies (i).

1.9. Comparability Graph

An undirected graph \( G (V, E) \) is a comparability graph, if there exists an orientation \((V, F)\) of \( G \) satisfying \( F \cap F^- = \emptyset \), \( F + F^- = E \), \( F^2 \subseteq F \); where \( F^2 = \{ (a, c) | (a, b), (b, c) \in F \text{ for some vertex } b \} \). The relation \( F \) is a strict partial ordering of \( V \) whose comparability relation is exactly \( E \), and \( F \) is called a transitive orientation of \( G \) (or of \( E \)). Comparability graphs are also known as transitively orientable graphs and partially orderable graphs.

The characteristics of comparability graphs were studied by P.C.Gilmore and A. J. Hoffman [GH 64].
Theorem 1.9.1 [GH 64]

A graph $G$ is a comparability graph if and only if each odd cycle has at least one triangular chord.

A **multigraph** is a graph with multiple edges between pairs of vertices. A **line graph** $L(M)$ of a multigraph $M$ has a vertex for every edge of $M$ with two vertices adjacent in $L(M)$ exactly when the corresponding edges in $M$ are adjacent. When $H = L(M)$ we say that $M = L^{-1}(H)$. A multigraph $M$ is bipartite if the vertices can be partitioned into two parts so that no two vertices in the same part are adjacent. A multigraph $M$ is **triangle free** if it does not contain three mutually adjacent vertices. A **star tree** is a connected bipartite graph $G$ with one part consisting of a single vertex.

1.10. **Matrix representation of a graph**

*Adjacency Matrices, Clique Matrices* etc. can be used to represent a graph. A weighted graph is represented by the **weight matrix** whose elements are the weights of the edges.

Let $G (V, E)$ be a graph whose vertices have been ordered arbitrarily as $v_1, v_2, \ldots, v_n$. Then the adjacency matrix $M = (m_{ij})$ of $G$ is an $n \times n$ matrix with entries $m_{ij} = 0$, if $(v_i, v_j) \notin E$, and $m_{ij} = 1$, if $(v_i, v_j) \in E$.

The maximal cliques - verses - vertices incidence matrix of an undirected graph is called a **clique matrix**, if all the maximal cliques are included.
1.11. Intersection Graphs

1.11.1. Definition [Ta 99].

A graph $G$ is called an intersection graph for a family $F$ of sets, if there exists a $1 - 1$ correspondence between the vertices of $G$ and the sets of $F$, such that, two distinct vertices are adjacent if and only if the associated sets intersect. i.e., an arbitrary pair $x_\alpha, x_\beta$ of vertices of $G$ are joined by an edge of $G$ if and only if $\alpha \neq \beta$ and $S_v \cap S_\beta \neq \phi$, where $S_\alpha$ and $S_\beta$ are the corresponding associated sets in $F$ and $\phi$ denotes the empty set.

1.11.2. C-Intersection Graph [Ta 99]

The C-intersection graph for a class C of sets, is any graph that arises from a collection of sets in C, when one associates to each of the sets a vertex and understands vertices to be adjacent precisely when their respective sets have nonempty intersection.

1.11.3. Intersection Number

The intersection number of a given graph $G$ is the minimum number of elements in a set $S$ such that $G$ is an intersection graph on $S$.

1.11.4. Intersection Representation

An intersection representation of a graph $G$ (V, E) is a function $f$ from $V(G)$, into a family of sets $F$ such that distinct vertices $v_1$ and $v_2$ of $V(G)$ are adjacent in $G$ precisely when $f(v_1) \cap f(v_2) \neq \phi$.

1.12. Intersection Graphs having Specific Topological Pattern

Let $F$ be any family of nonempty sets. The Intersection graph of $F$ is obtained by representing each set in $F$ by a vertex and connecting two vertices
by an edge if and only if their corresponding sets intersect. When $F$ is allowed to be an arbitrary family of sets, the class of graphs obtained as intersection graphs is simply all undirected graphs.

The problem of characterizing the intersection graphs of families of sets having some specific topological or other pattern is of great interest and frequently has real life application.

1.12.1. Intersection Graphs of Curves and Straight Lines in the Plane [EET 76]

Let $V$ be a finite set of curves in the plane. An undirected finite graph $G(V, E)$ is called the intersection Graph of curves in the plane, if $V$ is the set of $G$'s vertices and an edge $(a, b) \in E$ if and only if the curves $a$ and $b$ intersect. In case, if all the curves are straight line segments, the corresponding graph is an intersection graph of straight lines.

It was proved that, the set of intersection graphs is a proper subset of the set of all graphs. Intersection graphs have applications in circuit lay out techniques.

1.12.2 Permutation Graphs

The intersection graph of the line segments is called a permutation graph. Permutation graphs are a class of perfect graphs having large number of applications.

Also, If $\Pi$ is a permutation of the numbers $1, 2, 3, 4, \ldots, n$, then the graph $G[\Pi] = (V, E)$ is defined as, $V = \{1, 2, 3, \ldots, n\}$ and $ij \in E \iff (i - j) (\Pi_i^{-1} - \Pi_j^{-1}) < 0$, where $(\Pi^{-1})i = \Pi^{-1}$ denotes the position of the number $i$ in the sequence (permutation).
1.13. Intersection graphs of Subtrees in a Tree

Let $T$ be any tree and $F$ be any collection of subtrees of $T$. Two subtrees $T_1$ and $T_2$ in $F$ are said to have nonempty intersection if $V(T_1) \cap V(T_2) \neq \emptyset$. The intersection graph of subtrees in a tree is called a *subtree graph*.

Franica Gavril in his paper [Ga 74], has proved that the Intersection graphs of subtrees in Trees are exactly the Chordal Graphs. Triangulated graphs, which are the intersection graphs of subtrees in trees were proved choral.

Gavril in [Ga 74] proved the following theorem.

**Theorem 1.13.1**

A graph $G(V, E)$ is a subtree graph if and only if there exists a tree $T$ whose set of vertices is $\mu(G)$, the set of cliques, so that, for every $v \in V$, $T(\mu_v(G))$ where $\mu_v(G)$ is the set of cliques containing $v$, is connected.

A graph is called a *proper subtree* graph if it is the intersection of a family of subtrees of a tree so that no one of the subtrees is contained in another.

Regarding subtrees, Gavril proved:

**Theorem 1.13.2.** [Ga 74]

The following three conditions are equivalent

(i) $G$ is a subtree graph.

(ii) $G$ is a proper subtree graph.

(iii) $G$ is a chordal graph.
The pair \((T, T_{\mu}(G))\) satisfying theorem 1.13.1, is called \textit{clique tree representation} for the chordal graphs. This representation has motivated the researchers to study other types of intersection graphs.

We have the following theorem and lemma according to Seymour & Weaver and Jean R.S. Blair & Barry Peyton

\textbf{Theorem 1.13.3. [SW 84]}

If \(G\) is a chordal graph and \(v\) is a vertex of \(G\), then one of the following is true.

(i) \(v\) is adjacent to every other vertex of \(G\).

(ii) There is a separation \((V_1, V_2)\) of \(G\) such that \(v \in V_1 - V_2\) and \(G \setminus (V_1 \cap V_2)\) is a clique.

\textbf{Lemma 1.13.4 [BP 93]}

Every chordal graph \(G\) has a simplicial vertex. If \(G\) is not complete, then it has two nonadjacent simplicial vertices.

\textbf{1.14. Triangulated Graph [Chordal Graph]}

Every simple cycle of length strictly greater than 3 possess a chord. This property is called the \textit{triangulated graph property}. Graphs which satisfy this property are called \textit{triangulated graphs}. One of the first classes of graphs to be recognized as being perfect was the class of triangulated graph.

The interval graph constitute a special type of triangulated graph. Triangulated graphs have also been called \textit{Chordal, Rigid Circuit, Monotone Transitive, and Perfect Elimination Graphs}. 
A vertex \( v \) of \( G \) is called simplicial if its adjacency set \( \text{Adj}(v) \) induces a complete subgraph of \( G \); i.e., \( \text{Adj}(v) \) is a clique (not necessarily maximal).

Every triangulated graph \( G (V, E) \) has a simplicial vertex. Moreover, if \( G \) is not a clique, then it has two nonadjacent simplicial vertices [Go 80].

Triangulated graphs can be recognized in linear time.

A graph is triangulated, if and only if it is the intersection graph of a family of subtrees of a tree.

1.14.1. Helly Property [Go 80]

A family \( \{T_i\}_{i=1} \) of subsets of a set \( T \) is said to satisfy the Helly Property, if \( J \subseteq I \) and \( T_i \cap T_j \neq \emptyset \) for all \( i, j \in J \) implies that \( \bigcap_j T_j \neq \emptyset \).

We have the following proposition due to Golumbic,

**Proposition 1.14.2** [Go 80]

A family of subtrees of a tree satisfies the Helly Property.

1.14.3. Perfect Vertex Elimination Scheme [Procedure to recognize triangulated graphs]

Fulkerson and Gross suggested an iterative procedure to recognize triangulated graphs; viz., “repeatedly locate a simplicial vertex and eliminate it from the graph, until either no vertices remain and the graph is triangulated or at some stage no simplicial vertex exists and the graph is not triangulated” [FG, 65]. i.e., let \( G (V, E) \) be an undirected graph and let \( \sigma = [v_1, v_2, ..., v_n] \) an ordering of the vertices. Now, \( \sigma \) is called a perfect vertex elimination scheme or perfect scheme, if each \( v_i \) is a simplicial vertex of the induced
subgraph $G \{v_1, v_2, v_3, \ldots, v_n\}$. That is each $X_i = \{ v_j \in \text{Adj}(v_i) \mid j > i \}$ is complete.

**Theorem 1.14.4** [BP 93]

A graph $G$ is chordal if and only if $G$ has a perfect elimination ordering.

### 1.14.5. Vertex Separator [BP 93]

A subset $S \subset V$ is a vertex separator for nonadjacent vertices $a$ and $b$ (or $ab$-separator if the removal of $S$ from graph separates $a$ and $b$ into distinct connected components. Also if no proper subset of $A$ is an $ab$-separator, then $S$ is a minimal vertex separator for $a$ and $b$. Fulkerson and Gross proved that, every maximal clique is of the form $\{v\} \cup X_v$,

where $X_v = \{ x \in \text{Adj}(v) \mid \sigma^{-1}(v) < \sigma^{-1}(x) \}$, and $\sigma$ is a perfect elimination scheme for $G(V, E)$, a triangulated graph. Also they have proved the following proposition.

**Proposition 1.14.6.** [FG 65]

A triangulated graph on $n$ vertices has at most $n$ maximal cliques, with equality if and only if the graph has no edges.

**Theorem 1.14.7** [BP 93]

A graph $G$ is chordal if and only if every minimal vertex separator of $G$ is complete in $G.$

Since the triangulated graph is perfect, we produce both a stable set and clique cover of size $\alpha(G).$ The most important types of subtree graphs are the Interval graphs and the Circular – arc graphs.
1.15. Interval Graphs

An undirected graph $G$ is called an interval graph if its vertices can be put into one-to-one correspondence with a set of intervals $I$ of a linearly ordered set such that two vertices are connected by an edge of $G$ if and only if their corresponding intervals overlap at least partially or, have nonempty intersection.

If these intervals are of unit length, we get a unit interval graph. If no interval properly contains another, the graph constructed from that family of intervals on a line will be called a proper interval graph. It is well known that the classes of unit interval graphs and proper interval graphs coincide.

We list the following theorems and propositions characterizing Interval Graphs.

**Theorem 1.15.1** [Go 80]

Interval graphs can be recognized in linear time.

**Theorem 1.15.2**. [LB 62, HKM 82]

A finite graph is an interval graph if and only if, it is triangulated and has no asteroidal triples; where three vertices $u, v, w$ in a graph $G$ form an asteroidal triple if each pair of them is joined by a path which contains no neighbors of the third point [Ke 69, Go 80].

**Proposition 1.15.3.** [Go 80]

An induced subgraph of an interval graph is an interval graph.

**Proposition 1.15.4.**

An interval graph satisfies the triangulated graph property.
Theorem 1.15.5.

The complement of an interval graph satisfies the transitive orientation property.

**Theorem 1.15.6 [GH 64]**

An undirected graph $G$ is an interval graph if and only if $G$ is a triangulated graph and its complement $G^c$ is a comparability graph.

**Theorem 1.15.7 [GH 64]**

Let $G$ be an undirected graph. The following statements are equivalent.

(i) $G$ is an interval graph.

(ii) $G$ contains no chordless 4-cycles and its complement $G^c$ is a comparability graph.

(iii) The maximal cliques of $G$ can be linearly ordered such that, for every vertex $x$ of $G$, the maximal cliques containing $x$ occur consecutively.

Let $A$ be a $(0, 1)$ matrix. We say that $A$ has the *consecutive 1's property* (for columns) provided there is a permutation matrix $P$ such that the 1’s in each column of $PA$ occur consecutively [FG 65].

**Theorem 1.15.8. [GH 64]**

A graph $G$ is an interval graph if and only if every quadrilateral in $G$ has a diagonal and every odd cycle in $G^c$ has a triangular Chord.
Theorem 1.15.9 [HKM 82]

A graph $G$ is an interval graph if and only if it is chordal and contains no asteroidal triples.

The coloring, clique, stable set, and clique cover problems can be solved in polynomial time for interval graphs. The earliest characterization of interval graphs was obtained by Lekkerkerker and Boland [LB 62]. Their result embodies the notion that an interval graph neither branch into more than two directions, nor circle back into itself.

1.16. The Circular–arc graphs

The intersection graphs obtained from collections of arcs in a circle are called circular–arc graphs. If we straighten the arcs out to a line, the arc becomes an interval. Hence every interval graph is a circular–arc graph while the converse is false.

We call $G$ a proper circular arc graph if there exists a circular–arc representation for $G$ in which no arc properly contains another [Go 80].

Alan Tucker proved the following theorem.

Theorem 1.16.1 [Tu 80]

An undirected graph $G$ $(V, E)$ is a circular arc graph if and only if its vertices can be circularly indexed $v_1, v_2, \ldots, v_n$ so that for all $i$ and $j$,

$$(v_i, v_j) \in E \implies \text{Either } v_{i+1}, \ldots, v_j \in \text{Adj} (v_i)$$

$$\implies \text{or } v_{j+1}, \ldots, v_i \in \text{Adj} (v_j)$$

$$(\text{if } i < j, \text{ then } v_{j+1}, \ldots, v_i \text{ means } v_{j+1}, \ldots, v_n, v_1, \ldots, v_i).$$

A graph $G$ is called a Helly Circular-arc graph if there exists a circular-arc representation for $G$ which satisfies the Helly property.
The circular arc graphs have also been studied by Jeremy Spinrad [Sp 86] and Peter L. Renz [Re 70]. We have the theorem by Gavril, and a few
results in circular arc graphs.

Theorem 1.16.2. [Ga 74]

An undirected graph G is a Helly circular arc graph if and only if its
clique matrix has the circular 1’s property for columns.

Theorem 1.16.3. [Tu 80]

An undirected graph G is a circular arc graph if its augmented
adjacency matrix has the circular 1’s property for columns.

Theorem 1.16.4 [Tu 80]

An undirected graph G is a proper circular arc graph if and only if its
augmented adjacency matrix has the circular 1’s property for columns and for
every permutation of the rows and columns that is a cyclic shift or inversion of
their circular 1’s order, the last 1 in the first column does not occur after the
last 1 in the second column, excluding columns which are either all zeros or all
ones.

Theorem 1.16.5 [Tu 80]

If G is a proper circular -- arc graph, then G has a proper circular -- arc
representation in which no two arcs share a common end point and no two arcs
together cover the entire circle (that is they do not intersect at both ends).

1.17. Circle Graph

A graph is a circle graph if the vertices can be mapped to chords of a
circle, so that two vertices are adjacent if and only if the corresponding chords
of the circle intersect. An undirected graph $G$ is called a circle graph if it is isomorphic to the intersection graph of a finite collection of chords of a circle where no two chords share a common end point [Go 80]. The Circle graph is recognized in Polynomial time by Gabor and Supowit [GS 89].

1.18. **Overlap Graphs**

A graph $G$ is called an overlap graph if its vertices may be put into one-to-one correspondence with a collection of intervals on a line such that two vertices are adjacent in $G$ if and only if their corresponding intervals overlap (the two intervals intersect but neither properly contains the other).

**Proposition 1.18.1.** [Go 80]

An undirected graph $G$ is a circle graph if and only if $G$ is an overlap graph.

1.19. **Block Graphs** [Ha 69]

A cut point of a graph is one whose removal increases the number of components of the graph. Thus if $v$ is a cut point of a connected graph $G$, then $G - v$ is disconnected. A non separable graph is connected, nontrivial and has no cut points. A block of a graph is a maximal non separable subgraph.

If we take the blocks of $G$ as the family of sets, then the intersection graph $\Omega (F)$ is the block graph of $G$, denoted by $B(G)$. The blocks of $G$ correspond to the points of $B(G)$ and two of these points are adjacent whenever the corresponding blocks contain a common cut point of $G$. Clearly block graphs are subclass of chordal graphs.

Harary [HA 63] proved that, a graph $H$ is the block graph of some graph if and only if every block of $H$ is complete.
1.20. Path Graph

A path graph is the intersection graph of paths in a tree; taking the paths to be the set of edges making up the path we get the Edge Path Graphs and taking the paths to be the set of vertices making up the path, we get the Vertex Path Graphs.

Clyde L. Monma and Victor K. Wei have studied different classes of path graphs their tree representations, clique separator and have constructed recognition algorithms. The concept of the Perfect Vertex Path Graph (PV) and Compact Vertex Path Graphs (CV) were introduced by A. Antonysamy [An 94]. We discuss this more detail in Chapters V and VI.

1.21. Clique Separator [MW 86]

A clique C is a separator, if G (V \ C) is not connected. Let C separates G into Gi = G (C \ Vi), 1 ≤ i ≤ s. cliques which intersect C but not identical to C are called relevant cliques.

1.21.1 Atom [MW 86]

An atom is a connected graph with no separator.

Theorem 1.21.2 (Atom Theorem) [MW 86]

A chordal atom consists of either a single clique or two intersecting cliques.

Theorem 1.21.3. (Separator Theorem) [MW 86]

Assume, Clique C separates G, into subgraphs Gi = G (C \ Vi), 1 ≤ i ≤ s, s ≥ 2; then G is a chordal graph if and only if each Gi is chordal.
1.21.4. Antipodal Cliques and Antipodal graphs [MW 86]

Assume clique $C$ separates a path graph $G_i = G(C \cup V_i), 1 \leq i \leq s$. Cliques which intersect $C$ but are not identical to $C$ are called relevant.

Two cliques $C_1$ and $C_2$ are unattached, denoted by $C_1 \perp C_2$, if $(C_1 \cap C) \cap (C_2 \cap C) = \emptyset$. We say that $C_1$ dominated $C_2$, denoted by $C_1 \geq C_2$ if $(C_2 \cap C) \subseteq (C_1 \cap C)$. We say that $C_1$ Properly dominates $C_2$, denoted $C_1 > C_2$, if $C_1 \cap C \Rightarrow C_2 \cap C$. $C_1$ and $C_2$ are Congruent denoted $C_1 \sim C_2$ if $C_1 \geq C_2$ and $C_2 \geq C_1$. The two cliques are Antipodal, denoted by $C_1 \leftrightarrow C_2$ if they are attached and neither dominates the other. The dominance relation on cliques is transitive, that is, $C_1 \geq C_2$ and $C_2 \geq C_3$ imply $C_1 \geq C_3$. Also, two separated subgraphs $G_1 = G(C \cup V_1)$ and $G_2 = G(C \cup V_2)$ are unattached, denoted by $G_1 \perp G_2$, if $C_1 \perp C_2$ for every $C_1$ in $G_1$ and every $C_2$ in $G_2$. Other properties are derived similarly for separated graphs.

**Lemma 1.21.5.** [MW 86]

Two subgraphs $G_1$, and $G_2$ are antipodal if and only if

1. $C_1 \leftrightarrow C_2$,
2. $C_1 > C_2$, $C_1' < C_2'$
3. $C_1 > C_2$, $C_1' \geq C_2'$, $C_1' \cap C_2''$, (or $C_2 > C_1$, $C_2' \geq C_1'$, $C_2' \cap C_1''$)
4. $C_1 \sim C_2$, $C_1 \cap C_2'$, $C_2'' \sim C_1'$, $C_2'' \cap C_1''$, for some $C_1$, $C_1'$, $C_1''$ in $G_1$, $C_2$, $C_2'$, $C_2''$ in $G_2$. (These cliques need not all be different)

**Lemma 1.21.6.** [MW 86]

Let $C_1$ and $C_2$ be cliques in $G_1$ and $G_2$, respectively, with $G_1$ and $G_2$ not antipodal. If $C_1 > C_2$, then $G_1 > G_2$. 
Lemma 1.21.7. [MW 86]

A collection of pair wise non-antipodal subgraphs $G_i$ of a general graph $G$, can be arranged in such a way that $G_i \succ G_j$ implies $i < j$.

1.22. Bipartite Intersection Graphs [HKM 82]

Given a set $S$ and a family $F$ of distinct subsets of $S$, partition $F$ into two sub-families $F_1$ and $F_2$. The bipartite intersection graph of $F$ with respect to the given partition $(F_1, F_2)$ is the graph $G(V, E)$ with $V = F$ and $(F_i, F_j) \in E$ if $F_i \in F_1$ and $F_j \in F_2$ and $F_i \cap F_j \neq \emptyset$. That is $\Omega(F_1, F_2)$ is that graph obtained from the intersection graph $\Omega(F)$ by removing those edges between vertices in $F_1$, and between vertices in $F_2$.

1.22.1. Bi-Interval Graphs

Since the most intensively studied intersection graphs are the Interval graphs, its bipartite version is also important.

If $F$ is a family of intervals, partitioned into subfamilies $F_1$ and $F_2$, then the intersection graph over this partition, $\Omega(F_1, F_2)$ will be called a bi-interval graph.

1.22.2. Bi-Chordal graph

A bipartite graph will be called bi-chordal if it has no induced cycle of length greater than or equal to six.

1.22.3. Bi-asteroidal triple

A bi-asteroidal triple is a set of points $(u, v, w)$ of a bipartite graph such that between any pair of them, there exists a path which is not adjacent to any point in the neighborhood of the third point.
1.22.4. Link of an edge e, Link(e)

If e = (u, v), then consider G - (u, v). Thus when an edge is deleted, all edges adjacent to it are deleted as well. By link (e), we mean the sub graph induced by \( N_G(u) \cup N_G(v) - (u, v) \). An edge for which link (e) is complete bipartite it is called a *simplicial* edge.

We have the following theorem,

**Theorem 1.22.5**

A bipartite graph is a bi-interval graph, if and only if it is bi-chordal and contains no bi-asteroidal triples.

**Corollary 1.22.5.1**

A bipartite graph G, is a bi-interval graph, if and only if it does not contain as induced subgraph, any of the cycles \( C_n, n = 6 \).

**Theorem 1.22.5.2**

Every bipartite graph is a bi-subtree graph of some star \( K_{1,n} \).

1.22.6. Bi-Subtree Graphs

A *bi-subtree* graph is the bipartite intersection graph of subtrees of some tree. Even though a graph G, is a subtree graph if and only if it is chordal; a bi-chordal graph is not a bi-subtree graph.

1.22.7 Bisimplicial edge

An edge e = (x, y) of a bipartite graph H (U, E) is *bisimplicial* if \( \text{Adj}(x) + \text{Adj}(y) \) induces a complete bipartite subgraph of H.
More discussion of the intersection graphs can be seen in [Ga 74, Ga 75, Ga 78, Go 80, Go 85, Re 70, Mo 86, EGLP 66] and the recognition of random intersection graphs is dealt by Erich Prisner [Pr 2000].

We give a tabular representation of the different families of intersection graphs discussed so far.

1.22.8. Diagram I

<table>
<thead>
<tr>
<th>GRAPH</th>
<th>FAMILY</th>
<th>TYPE OF INTERSECTION</th>
<th>AUTHOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topological</td>
<td>Curves or Straight lines In the plane.</td>
<td>Curve intersection or line intersection</td>
<td>Elrich, Even, Tarjan, Golumbic,</td>
</tr>
<tr>
<td>Permutation</td>
<td>Line segments</td>
<td>Line segment intersection</td>
<td>Golumbic</td>
</tr>
<tr>
<td>Chordal</td>
<td>Subtrees of a tree</td>
<td>Edge or vertex intersection</td>
<td>Gavril, Bang, Walter, Buneman, Surany, Berge, Antonysamy, Chandrasekaran, Iyengar, Blair, Peyton, Edenbrandt, Farber.</td>
</tr>
<tr>
<td>Interval</td>
<td>Intervals on a linearly ordered set</td>
<td>Interval intersection</td>
<td>Keil, Pandurangan, Antonysamy, Fishburn, Capobianco, Mollaro, Fulkerson, Gross, Gilmore, Hoffman, Chvatal, Klee, Kendall, Lekkerkerker, Boland</td>
</tr>
<tr>
<td>Proper Interval</td>
<td>Unit intervals on real line</td>
<td>Interval intersection</td>
<td>Rose, Tarjan, Korte, Booth, Golumbic</td>
</tr>
<tr>
<td>Circular Arc</td>
<td>Arcs on a circle</td>
<td>Arcs intersection</td>
<td>Golumbic, Hoffman, Gilmore, Tucker, Gavril, Spinrad, Klee, Rao, Pandurangan, Hsu, Tsai</td>
</tr>
<tr>
<td>Overlap</td>
<td>Intervals on a line</td>
<td>Intervals overlap</td>
<td>Golumbic</td>
</tr>
<tr>
<td>Circle</td>
<td>Chords of a circle</td>
<td>Chords intersection</td>
<td>Golumbic, Hsu, Sprinrad, Gabor, Supowit</td>
</tr>
<tr>
<td>Path</td>
<td>Paths in an undirected tree</td>
<td>Edge or vertex intersection</td>
<td>Golumbic, Jamison, Monma, Wei, Antonysamy.</td>
</tr>
<tr>
<td>Directed Path</td>
<td>Paths in a directed tree</td>
<td>Edge or vertex intersection</td>
<td>Golumbic, Jamison, Monma, Wei, Antonysamy,</td>
</tr>
<tr>
<td>Block Graph</td>
<td>Blocks of a graph</td>
<td>Intersection at a cutpoint</td>
<td>Harary</td>
</tr>
</tbody>
</table>
1.23. Complexity of Computer algorithms

1.23.1. Introduction

The Complexity analysis of Computer Algorithms is a major area in Computer Science. The time needed by an algorithm expressed as a function of the size of a problem is called the time complexity of the algorithm. Computability and Computational Complexity are the pillars of the theory of computation. Computability deals with the question, whether there is an algorithm which solves the problem $Q$, while the computational complexity deals with the quantitative aspects of problem solving. The issue in question is what can be computed within a practical or reasonable amount of time and space by measuring the resource requirements exactly or by obtaining the upper and lower bounds. The complexity is actually determined on three levels:

(i) the problem

(ii) the algorithm

(iii) the implementation

Hence we need the best algorithm which solves the problem and we want to choose the best implementation of that algorithm.

1.23.2. Algorithmic Complexity

We use the term algorithm to mean a set of well defined rules or instructions for obtaining a specific output from a specific input in a finite number of steps.

The complexity of an algorithm measures the amount of computational efforts extracted when the computer solves its problem using that algorithm.
This measure may refer to the number of computational steps, running time, or storage space. The complexity of an algorithm is commonly a function of the size and presentation of the input data. We associate with a problem, an integer, called the size of the problem which is a measure of the quantity of input data. The size of a graph problem might be the number of edges.

A Polynomial time algorithm is defined to be one whose time complexity function is $O(f(n))$ for some polynomial function $f$, where $n$ is used to denote the input length. If $T(n)$ is the time for an algorithm on $n$ inputs, then, we write $T(n) = O(f(n))$ to mean that the time is bounded above by the function $f(n)$, and $T(n) = \Omega(g(n))$ to mean that the time is bounded below by the function $g(n)$. Any algorithm whose time complexity function cannot be so bounded is called Exponential time algorithm.

1.23.3. Efficient algorithm or Fast algorithm

An algorithm is efficient or fast if its complexity is a polynomial in the input size $n$, of the data.

A computational problem is called tractable if there exists an efficient algorithm for solving the problem. A computational problem is intractable if it can be established that there is no algorithm to solve the problem. Or, a problem is intractable if all algorithms to solve the problem are of at least exponential time complexity. The implication of this rating scheme is that problems having polynomial time bounded algorithms are tractable.

1.23.4. Search Algorithms

Given an alphabetical listing of words, we wish to determine if and where a specific word appears on the list. A simple way is to start at the
beginning of the list and to search in sequence until the name is found or until we come to the end of the list. This algorithm is often called *sequential search algorithm*. If there are $n$ words on the list and if we want to know the position of a particular word then it will take $n$ comparisons before the algorithm terminates. Hence the complexity is measured in terms of the number of comparisons rather than directly as time units. Then the *worst case complexity of this algorithm is $O(n)$*. For a sequential search algorithm it is not necessary that the words have to be in alphabetical order.

1.23.5. Sorting Algorithms

In the search algorithms we assume that the given list is in alphabetical order but the sorting algorithms are useful to arrange a given list of words in alphabetical order. There are two algorithms to do this job, one is called sequential sort or selection sort algorithm and the other is called the *Bubble Sort Algorithm* or exchange sort algorithm. The worst case complexity of this algorithm is $O(n \log n)$.

1.23.6. NC Algorithms

There has been a considerable interest in problems which have parallel algorithms running in time bound by a polynomial in the logarithms of the size of the input and using a number of processors polynomial in the input size. Such a class of problems referred to as the class *Non Deterministic Polynomial Complete* (NC). There is a greater amount of interest in the NC because the speed of the NC algorithms is exponential compared to their sequential versions. Furthermore, they make use of a reasonable amount of hardware, namely, the processors. The class NC is ‘robust’ under reasonable
changes in the underlying machine models of parallel computation. Recently, NC algorithms have been given for recognizing comparability graphs, permutation graphs, and interval graphs [KVV 85, HM 86]. S. S. Iyengar’s NC algorithm [CI 88] for recognizing chordal graphs and k-trees are of greater importance in the research area. These algorithms use $O(n^4)$ processors and take $O(\log n)$ time on the PRAM model of computation.

1.23.7. NP – Completeness

Class P is defined to be the set of problems which can be solved by algorithms of polynomial time complexity. Class NP is defined to be the set of problems for which possible solutions can be verified in polynomial time. Algorithms for solving problems in this class are known as ‘Nondeterministic’ algorithms, and it consists of two stages: the ‘guessing’ stage in which the possible solutions are made available, and the ‘checking’ stage in which it is determined whether each such possibility is in fact a solution. For NP problems the checking stage is required to be polynomially bounded. The term ‘NP’ represents the capacity to solve problems in polynomial time on a nondeterministic Turing Machine, as discussed by Garay & Johnson [GJ 79], Aho, Hopcroft and Ullman [AHU 2001] and many others.

Every problem in class P is also in NP, that is, $P \subseteq NP$. This is known by observation that any of the algorithms in P can be used as the checking stage of some nondeterministic algorithms [GJ 79]. It is not known whether there are problems in NP which are not in P. The issue of whether $P = NP$ remains an open question, although it is widely believed that $P \subseteq NP$. 
A large number of problems in NP, (there are well over 300 listed in [GJ 79]), have been proved equivalent in terms of complexity in the sense that if one of these problems is in P then they are all in P. These problems are said to be NP – Complete. This equivalence is based on polynomial time reducibility, whose problem L is reducible to problem M if every instance of problem L can be uniquely transformed by an algorithm of polynomial complexity into an instance of M. A more precise definition of class NP – complete is that it is the set of problems L such that L is in NP, and all other problems in NP can be polynomial time reduced to L.

The importance of class NP – Complete is as follows: Development of a polynomial algorithm for any NP – complete problem would imply existence of polynomial algorithms for all other NP – complete problems, and would prove that P = NP. If any problem in NP is intractable, then all NP – complete problems are intractable. If the dominant belief that P ≠ NP is correct, then problems must exist which are in NP, but are neither in P nor NP – complete [GJ 79].

* * * * *