CHAPTER 1

INTRODUCTION

Graphs are used to represent a number of real-life situations. Transportation systems, communication systems, job scheduling, and facility supply problems are some useful areas, which are represented and analyzed using graph models. Even social applications, such as personal relationship among the staff of an institution can be represented and studied using graphs. After the invention of computers, the study of graph theory got accelerated and the algorithmic graph theory was rigorously studied. Efficient algorithms for various graph problems have been developed.

In recent years, the design and analysis of parallel algorithms for graph problems are considered to be a very useful area of research. In this chapter we give the basic definitions and theorems of graph theory that are needed for the subsequent chapters. We also introduce the centrality concepts, and define central structures. For graph theoretic terminology, we refer to Harary [15] and Parthasarathy [22].
1.1 Graph Terminologies

**Definition 1.1.** A graph $G$ is a finite non-empty set of objects called vertices together with a set of unordered pairs of distinct vertices of $G$, called edges. The vertex set and the edge set of $G$ are denoted by $V(G)$ and $E(G)$ respectively.

If $e=\{u,v\}$ is an edge, we write $e=uv$; we say that $e$ joins the vertices $u$ and $v$; $u$ and $v$ are adjacent vertices; $u$ and $v$ are incident with $e$.

If two vertices are not joined, then we say that they are non-adjacent. If two distinct edges are incident with a common vertex, then they are said to be adjacent to each other.

**Definition 1.2.** The degree of a vertex $v$ in a graph $G$ is the number of edges of $G$ incident with $v$ and is denoted by $\deg v$. A vertex of degree 0 in $G$ is called an isolated vertex; a vertex of degree 1 is called a pendant vertex or an end vertex of $G$. Any vertex which is adjacent to a pendant vertex is called a support. A vertex of odd degree is an odd vertex and a vertex of even degree is an even vertex.

**Theorem 1.3 [22].** The sum of the degrees of a graph is even, being twice the number of edges.
Theorem 1.4 [22]. The number of odd vertices in a graph is even.

Definition 1.5. A graph $G$ is complete if every pair of its vertices is adjacent. A complete graph on $p$ vertices is denoted by $K_p$.

Definition 1.6. Let $u$ and $v$ be (not necessarily distinct) vertices of a graph $G$. A $u$-$v$ walk of $G$ is a finite, alternating sequence $u = u_0, e_1, u_1, e_2, \ldots, e_n, u_n = v$ of vertices and edges beginning with vertex $u$ and ending with vertex $v$ such that $e_i = u_{i-1}u_i, i=1, 2, \ldots, n$. The number $n$ is called the length of the walk. The walk is said to be open if $u$ and $v$ are distinct vertices; it is closed otherwise. A walk $u_0, e_1, u_1, e_2, \ldots, e_n, u_n$ is determined by the sequence $u_0, u_1, u_2, \ldots, u_n$ of its vertices and hence we specify this walk by $(u_0, u_1, u_2, \ldots, u_n)$.

Definition 1.7. A path in a graph is an open walk in which no vertex (and therefore no edge) is repeated.

Definition 1.8. A trail in a graph is a walk in which no edge is repeated. A tour is a closed trail.

Definition 1.9. A closed walk in which no vertex (and edge) is repeated is called a cycle.
Definition 1.10. Two distinct vertices $u$ and $v$ of a graph $G$ are said to be connected (or joined) if there is a $u$-$v$ walk in $G$. By convention, a vertex is connected to itself. A graph $G$ is said to be connected if every two of its vertices are connected; otherwise it is disconnected.

Definition 1.11. A maximal connected subgraph of $G$ is called a component of $G$.

Definition 1.12. A graph $G$ is acyclic if it has no cycles. An acyclic graph is also called a forest. A connected acyclic graph is called a tree.

Definition 1.13. A tree, which yields a path when its pendant vertices are removed, is called a caterpillar. A spider is a tree which has at most one vertex of degree $\geq 3$.

1.2 Centrality Concepts

The areas in which the concept of centrality in graphs is widely applied are facility location problems and social networks. Facility location problems arise from several real-life situations wherein we deal with the question of what is an optimal location for a facility in a graph.
In almost all cases the type of facility to be established is one for which a "central" structure is optimal.

Facility location problems deal with the task of choosing a site subject to some criterion. Many problems of finding the best site for a facility in a graph or network are in one of the following two categories:

a. **Minimax location problems**

For example, in locating an emergency response facility such as a fire station, police station or hospital, the primary concern may be to minimize the maximum distance from the location of facility to the vertex/node farthest from it.

b. **Minisum location problems**

For example, in locating a warehouse, one may be interested in minimizing the sum of the distances from the facility location to all the other vertices/nodes.

In 1948, Bavelas [3] introduced the idea of centrality as applied to human communication. Subsequent studies in such areas as sociology, geography, history and political science have been extensive. Pitts [23] has stated "... it is clear that several traditions within sociology and
social science are finding a social networks framework useful ... Accessibility, Betweenness and Centrality are now the ABCs of our vocabulary”. Freeman [7] describes “betweenness” as a “measure of overall centrality”.

Interestingly, it has been the sociologists, geographers, historians, etc, as opposed to the mathematicians and operations researchers, who have been primarily interested in the concept of theory of centrality.

This thesis focuses its attention on design and analysis of parallel algorithms for the facility location problems.

1.3 Central Structures

From problems involving the location of security and/or fire protection forces in a network, it was observed that such forces did not always have to respond to a single point in the network, but rather might be able to handle an emergency by responding to any point in some designated area about the critical point.

For other location theory problems, the nature of the facility to be constructed (such as a pipeline) could necessitate selecting a structure (such as a path) rather than just a point at which to locate the facility. Also existing structures (such as highways or bus routes) relative to which a new structure (such as an additional bus route) is to be located
might not occupy single points. This section gives the definitions and theorems for such central structures.

**Definition 1.14.** The length of any shortest path between \( u \) and \( v \) of a connected graph \( G \) is called the *distance* between \( u \) and \( v \) and is denoted by \( d(u,v) \).

**Definition 1.15.** Let \( G \) be a connected graph. For any vertex \( v \) of \( G \), the *eccentricity* of \( v \) is \( e(v) = \max \{ d(v,u) \mid u \in V(G) \} \). The *radius* of \( G \) is defined by \( r = \min \{ e(v) \mid v \in V(G) \} \) and the *diameter* of \( G \) is defined by \( \text{diam} = \max \{ e(v) \mid v \in V(G) \} \). A vertex \( u \) of \( G \) such that \( d(u,v) = e(v) \) is called an *eccentric vertex* of \( v \).

The sequence \( e_i \), where \( e = e(v) \) are the eccentricities of the vertices of \( G \), is called the *eccentricity sequence* of \( G \). The set of distinct elements in the eccentricity sequence is called the *eccentricity set* of \( G \).

**Definition 1.16.** Let \( G \) be a connected graph. Then the *center* of \( G \) is defined by \( C(G) = \{ v \in V(G) \mid e(v) = r \} \) and the *periphery* of \( G \) is defined by \( P(G) = \{ v \in V(G) \mid e(v) = \text{diam} \} \). A vertex in \( C(G) \) is called a *central vertex* and a vertex in \( P(G) \) is called a *peripheral vertex*. A graph with \( C(G) = V(G) \) is called a *self-centered graph*. 

7
**Theorem 1.17 [22].** The radius and diameter of a graph are related as follows:

\[ r \leq \text{diam} \leq 2r. \]

**Theorem 1.18 [15].** Every nontrivial tree has at least two endpoints.

**Theorem 1.19 [15].** Every tree has a center consisting of either one point or two adjacent points.

**Definition 1.20.** A branch at a point \( u \) of a tree \( T \) is a maximal subtree containing \( u \) as an endpoint. Thus the number of branches at \( u \) is \( \text{deg} \ u \).

**Definition 1.21.** The branch weight, \( bw(u) \), at a point \( u \) of \( T \) is the maximum number of lines in any branch at \( u \).

**Definition 1.22.** A point \( v \) is a centroid point of a tree \( T \) if \( v \) has minimum branch weight, and the centroid of \( T \) consists of all such points.

**Theorem 1.23 [15].** Every tree has a centroid consisting of either one point or two adjacent points.
Definition 1.24. Let $G$ be a connected graph. For any vertex $v$ of $G$, the distance of $v$ is defined by $d(v) = \sum_{u \in V} d(v,u)$. The median of $G$ is defined by $m = \min \{ d(v) \mid v \in V(G) \}$. The subgraph induced by the set $M$ of median vertices of $G$ is called the median subgraph of $G$.

Theorem 1.25 [16]. Each tree contains either one or two medians, and if two then they are adjacent.

Definition 1.26 [25]. Let $G$ be a connected graph. For a subset $S \subseteq V(G)$ and any vertex $v \in V(G)$, let $d(v,S) = \min_{u \in S} d(v,u)$. Then for any $S \subseteq V(G)$,

Eccentricity $e(S) = \max_{v \in V(G)} d(v,S)$,

Distance $d(S) = \sum_{v \in V(G)} d(v,S)$, and

Branch weight $bw(S) = \max \{ |C| : C \text{ is the vertex set of a component of } G-S \}$.

Definition 1.27 [25]. Let $P$ be a path in a graph $G$ with $e(P) \leq e(P')$ for any path $P'$ in $G$ then. The path $P$ is called a path center of $G$ if there does not exist a path $P'$ in $G$ with fewer vertices than $P$ for which $e(P') = e(P)$.

Definition 1.28 [25]. Let $P$ be a path in graph $G$ with $bw(P) \leq bw(P')$ for any path $P'$ in $G$. The path $P$ is called a spine or path centroid of $G$ if
there does not exist a path $P'$ in $G$ with fewer vertices than $P$ for which $bw(P') = bw(P)$.

**Definition 1.29** [25]. A path of minimum distance is called a core or path median of $G$. Thus a path $P$ is a core if $d(P) \leq d(P')$ for every path $P'$ in $G$.

**Definition 1.30** [2]. The core of a graph $G$ is a path $P$ in $G$ that is central with respect to the property of minimizing $d(P) = \sum d(v, P)$, where $d(v, P)$ is the distance of vertex $v$ from $P$.

Let $T=(V,E)$ be a tree with each edge $(u,v) \in E$ associated with length $l(u,v)$. Let $P_{r,v}$ be a path from root $r$ to vertex $v$. Let $Q$ be a path, where $Q$ intersects $P_{r,v}$ at $r$. Let $d(Q) = \sum_{v \in V} d(v, Q)$, where $d(v, Q)$ is the distance of vertex $v$ from $Q$. Let $\text{reduction}(r, v)$ be the total reduction from $d(Q)$ when the path $Q$ ending at $r$ is extended along the path $P_{r,v}$.

**Lemma 1.31** [2]. Let $R$ be a path ending at $r$. The total reduction from $d(R)$, when $R$ is extended to include the vertices in $P_{r,v} - \{r\}$ is

$$\text{reduction}(r, v) = \sum l(u, w) \ast \text{size}(T_w); w \in P_{r,v} - \{r\}$$

where, $u$ is the parent of $w$ and $\text{size}(T_w)$ is the number of vertices in the subtree of $T$ rooted at $w$. 
Definition 1.32. Vertex $v$ is called an endpoint of a core if $\text{reduction}(r,v) \geq \text{reduction}(r,w)$ for all vertices $w$ in $T$.

Theorem 1.33 [26]. For any tree $T$, the path center is unique and contains the vertex center of $T$.

Theorem 1.34 [26]. For any tree $T$, the spine of $T$ is unique and contains vertex centroid of $T$.

Theorem 1.35 [25]. Any two cores of a tree $T$ have nonempty intersection.

Theorem 1.36 [25]. For any tree $T$ there is at least one vertex common to all the cores.

1.4 Tree representation

Since our main emphasis is on the central structures of a tree, its memory representation is explained here. We adopt a simple and straightforward method to represent a tree $T$ rooted at $r$. A parent array
$p$ is used to represent a tree, where $p(v)$ is the parent of vertex $v$, and $p(r) = 0$. Consider the tree given in figure 1.1 rooted at vertex 1.

![Figure 1.1 A rooted Tree](image)

Its array representation is given below

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

Even though the example given here is a binary tree, same representation holds good for a general tree also. If the tree $T$ contains $n$ vertices then a random numbering form 1 to $n$ may be used to label the vertices. The main operation that is needed on the tree is the traversal from a vertex $v$ to the root $r$, which can be done easily with the help of the parent array by using the pointer jumping technique. In the subsequent chapters we use the parent array to represent trees.