# On Gracefully Labeling Trees

Dhananjay P. Mehendale
Sir Parashurambhau College, Pune-411030, India.

## Abstract

A method to obtain all possible graceful spanning trees in a complete graph is proposed. An algorithm to generate all the labeled spanning trees in a complete graph is developed and modified to generate all graceful spanning trees. The count of all possible graceful graphs in a complete graph is obtained. An upper bound on the count of gracefully labeled trees in a complete graph is obtained. We settle Graceful Tree Conjecture in the affirmative in two ways: 1) We show that all trees can be gracefully labeled by assigning the lowest label 1 to the so called special vertices of trees, i.e. prependant vertices or pendant vertices adjacent to prependant vertices. 2) We establish the existence of graceful labeling for all trees by associating distinct lattice paths with trees and by showing the existence of a lattice path for a tree of each isomorphism type by showing how to construct a lattice path recursively by starting from the lattice path for its pendant vertex deleted subtree, which is assumed to exists by induction, and carrying out appropriate modification of this lattice path. Lastly, we discuss an algorithm to find arbitrarily degree constrained graceful spanning tree and propose some problems for further investigation.

**1. Introduction:** A tree on n vertices is said to be graceful or said to have a graceful labeling if when its vertices are labeled with integers {1, 2, …, n} and lines (edges) are labeled by the difference of their respective end vertex labels then all the edge labels taken together constitute the set {1, 2, …, n−1}.

In the year 1964 Ringel [1] proposed the following

**Conjecture 1.1(Ringel):** If T is a fixed tree with m lines, then $K_{(2m+1)}$, the complete graph on $(2m+1)$ vertices, can be decomposed into $(2m+1)$ copies of T.

Attempts to prove Ringel's conjecture have focused on a stronger conjecture about trees [2], called the Graceful Tree Conjecture:

**Conjecture 1.2 (Graceful Tree Conjecture):** Every (unlabeled) tree is graceful, i.e. has a graceful labeling.

As an implication of the validity of this conjecture for a particular tree T under consideration Rosa [3] proved the following

**Theorem 1.1(Rosa):** If a tree T with m lines has a graceful labeling then $K_{(2m+1)}$ has decomposition into (*2m+1*) copies of T.

Thus, the validity of Ringel's conjecture will automatically follow from the validity of the graceful tree conjecture.

In this paper, we start with an algorithm to generate all the labeled spanning trees in a complete graph on n vertices. This algorithm can also be used for generating all the labeled spanning trees in any (p, q) graph.

By doing a simple modification in this algorithm we will obtain the list of all the (labeled) graceful spanning trees that can exist in a complete graph on n vertices under consideration. We then proceed with the counting and generation of all spanning graphs on n vertices (and clearly on (n – 1) lines) that are graceful. We then give an estimate about the count of the graceful trees in a complete graph on n vertices. We will then proceed to produce and list, as an example, all the possible graceful trees for number of vertices equal to five (i.e. n = 5).

Finally we proceed with the settlement of graceful tree conjecture in the affirmative, by making use of the theorem about recursive generation of trees by extending at "special vertices" of the trees which are lower in size by one vertex.

**2. Preliminaries:** Let G be a simple connected (n, q) graph with the vertex set $V(G) = \{v_1, v_2, \ldots, v_n\}$ and edge set $E(G) = \{e_1, e_2, \ldots, e_q\}$. Let the matrix $A(G) = [x_{ij}]$ denotes the reduced incidence matrix obtained by deleting the $n^{th}$ row, corresponding to vertex $v_n$, from the incidence matrix of G.

**Lemma 2.1:** A square submatrix $M$ of $A(G)$ of size (n−1) is nonsingular if and only if its determinantal expansion contains a unique nonzero monomial.

**Proof:** We have rank of $A(G)$ = n – 1 and every column of $A(G)$ contains at most two nonzero entries (units). If $M$ be nonsingular, then there must be a column in $M$, say $j_1$, which contains exactly one nonzero entry, a unit, in the row, say $i_1$ (since, if every column of $M$ will contain two nonzero entries (units) then addition of all rows of $M$ modulo 2 will produce a row of all zeroes contradicting the nonsingularity of $M$). Expanding by this column the determinant of $M$, $Det(M)$, will be

$$Det(M) = x_{i_1 j_1} Det(B_{i_1 j_1})$$

Now, the nonsingularity of $M$ implies the nonsingularity of the cofactor matrix $B_{i_1 j_1}$ and it also in return should contain a column say $j_2$, containing only one nonzero value, a unit, in the row, say $i_2$. Further expanding by this column we will have

$$Det\,(M\,) = x_{i_1 j_1} x_{i_2 j_2} Det\,(C_{i_2 j_2})$$

where $C_{i_2 j_2}$ is the corresponding cofactor matrix.

Now, by repeated application of this procedure which is assured by the nonsingular nature of $M$ and which terminates in (n─1) steps we will get

$$Det(M) = \pm x_{i_1 j_1} x_{i_2 j_2} \cdots x_{i_{(n-1)} j_{(n-1)}}$$

where $x_{i_1 j_1} x_{i_2 j_2} \cdots x_{i_{(n-1)} j_{(n-1)}}$ is the desired unique nonzero monomial. The converse straightforwardly follows from the existence of the unique nonzero monomial which implies that $Det(M) \neq 0$ and so $M$ is nonsingular. Hence the lemma.

□

Note that the content of the first index set of all the nonzero monomials in $A(G)$ is identical, namely, {1, 2, …, n─1} and they differ only in the content of the second index set which could be $\{t_1, t_2, \cdots, t_{(n-1)}\}$, such that $1 \leq t_1 < t_2 < \cdots < t_{(n-1)} \leq q$. Thus, a monomial is a unique nonzero monomial when it is the only nonzero monomial with the given content of the second index set. Thus, when there is more than one nonzero monomial with a same content for the second index set in a maximal square matrix of $A(G)$ the matrix will be singular.

The following theorem [4] describes the one-to-one correspondence between the spanning trees in a graph and the nonsingular matrices of $A(G)$ of size (n – 1):

**Theorem 2.1:** A matrix of $A(G)$ of size (n – 1) is nonsingular if and only if the (n─1) edges corresponding to the (n─1) columns of this matrix constitute a spanning tree in G.

**Definition 2.1:** The nonzero (determinantal) monomials generated by selecting nonzero entries in succession from top row to bottom row and from every possible columns of $A(G)$ are called the **nonzero monomials** of $A(G)$.

The lemma 2.1 and the theorem 2.1 together imply the following

**Theorem 2.2:** There is a one-to-one correspondence between the uniquely occurring (i.e. occurrence of only one nonzero monomial having the associated content of the second index) nonzero monomials in $A(G)$ and the labeled spanning trees in G. Moreover, a uniquely occurring monomial with the content of the second index set $\{t_1, t_2, \cdots, t_{(n-1)}\}$, such that $1 \le t_1 < t_2 < \cdots < t_{(n-1)} \le q$ constitutes a spanning tree in G with edge labels $\{ e_{t_1}, e_{t_2}, \cdots, e_{t_{(n-1)}} \}$.

**Definition 2.2:** A nonzero monomial in G is called **nonunique** if it occurs more than once having the same content for the second index set during the generation procedure of all the nonzero monomials of $A(G)$.

The proof of the following theorem is clear from the characterization of a tree:

**Theorem 2.3:** The nonunique (one with repeated occurrence) monomial of $A(G)$ represents a disconnected (n, n─1) graph of G containing at least one circuit (cycle).

We are now ready to present an algorithm to generate all the labeled spanning trees in a graph:

**Algorithm 2.1:**

(1) Generate the entire column content vectors by selecting the column labels of $A(G)$ as components in succession corresponding to a nonzero entry from each row from top row to bottom row.
(2) Arrange the components of all these vectors in increasing order.
(3) The list of all uniquely occurring columns content vectors represent all the labeled spanning trees with edge labels associated with the column labels in the column content vectors.
(4) The list of all columns content vectors with repeated occurrence represent all the labeled disconnected (n, n─1) graphs with edge labels associated with the column labels in the column content vectors.

**3. Generation of Gracefully labeled Trees:**

In this section we describe a procedure for generating all the graceful trees in a complete graph on n vertices. The procedure is similar to the procedure discussed above for generating all the labeled spanning trees. The only change is an additional constraint during the formation of all the column content vectors, as per step (1) in algorithm 2.1, so that the generated labeled spanning graphs (either a tree or a disconnected graph, decided in further steps of algorithm 2.1) are graceful. Few definitions are in order:

Let G be a simple connected (n, q) graph and $A(G) = [x_{ij}]$ its reduced incidence matrix.

**Definition 3.1:** The matrix $P(G) = [y_{ij}]$ will denote the matrix of the same size of $A(G)$ obtained by replacing the entries $x_{ij}$ of $A(G)$ by a pair $(i, j)$ when $x_{ij} = 1$.

**Definition 3.2:** The matrix $N(G) = [z_{ij}]$ will denote the matrix of the same size of $P(G)$ obtained by replacing every pair entry (i, j) in $P(G)$ by the difference $|i - j|$.

**Definition 3.3:** The vector formed by selecting one entry from each row of $N(G)$ as a component is called **improper choice vector** if the selected components together, after omitting the repetitions, form a **proper subset** of the set {1, 2,…, (n−1)}. i.e. there is a repetition of some edge label.

**Definition 3.4:** The vector formed by selecting one entry from each row of $N(G)$ as a component is called **proper choice vector** if the selected components together form the set {1, 2,…, (n−1)}.

**Definition 3.5:** The vector formed by selecting from $P(G)$ the proper vertex-pairs as components from the same row corresponding to the components of proper choice vector is called the **corresponding vertex-pair vector** (i.e. the vector whose components are vertex pairs).

We now proceed to generate all the graceful spanning trees in $K_n$ through the following

**Algorithm 3.1:**

(1) Form the matrices $A(K_n), P(K_n)$ and $N(K_n)$.

(2) Generate all the proper choice vectors, by selecting entries, one entry from each row of $N(K_n)$ from top row to bottom row, as components for these vectors.
(3) Form corresponding vertex-pair vectors by selecting corresponding vertex pairs as entries, one entry from each row of $P(K_n)$ from top row to bottom row, as components for these vectors.

We now proceed to give the exact count of the all possible graceful spanning subgraphs that can exist in $K_n$ and also give a simple procedure for their generation. Also, we find an estimate for the count of the all possible graceful spanning trees that can exist in $K_n$, a complete graph on n vertices.

**Theorem 3.1:** There are in all (n─1)! spanning subgraphs in $K_n$ that are graceful.

**Proof:** We form graceful spanning subgraphs by selecting all possible distinct sets of vertex-pairs (i.e. differing with respect to at least one vertex-pair) like:

$$\{(i_n, j_n), (i_{(n-1)}, j_{(n-1)}), (i_{(n-2)}, j_{(n-2)}), \ldots, (i_2, j_2)\}$$

such that:

$$(i_n, j_n) \in \{(1, n)\}$$
$$(i_{(n-1)}, j_{(n-1)}) \in \{(1, n-1), (2, n)\}$$
$$(i_{(n-2)}, j_{(n-2)}) \in \{(1, n-2), (2, n-1), (3, n)\}$$
$$\vdots$$

$(i_2, j_2) \in \{(1, 2), (2, 3), (3, 4), \cdots, (n-1, n)\}$ and **view it as a subgraph of $K_n$.**

The first pair $(i_n, j_n)$ has only one choice, the second pair has two choices, …, the last pair has (n─1) choices and all these choices can be made independently. Also, any two constructions of a set of vertex-pairs like:

$$\{(i_n, j_n), (i_{(n-1)}, j_{(n-1)}), (i_{(n-2)}, j_{(n-2)}), \ldots, (i_2, j_2)\} \rightarrow (*)$$

represent distinct graceful graphs when the sets differ at some one (or more) elements. Also, every graceful spanning graph gets generated in this way, because if we consider any graceful spanning graph and form its edge set in terms of the vertex-pairs then clearly it will be some set of the type (*), among the sets generated above. Hence, the result.

□

Let $\alpha = \left[\frac{n}{2}\right]$, where [s] denotes the integral part of s.

**Theorem 3.2:** The total count of **graceful** spanning trees in $K_n$ is $\leq$ C, where

C = (1.2.3. … $(\alpha-1).\alpha.\alpha.(\alpha-1)\ldots$3.2.1).2$^{(\alpha-1)}$, when n is odd, and

C = (1.2.3. … $(\alpha-1).\alpha.(\alpha-1)\ldots$3.2.1).2$^{(\alpha-1)}$, when n is even.

**Proof:** It is easy to check from $N(K_n)$ that all entries {1, 2, 3, …, (n–1)} appear only once in first row. Entry 1 appears twice while entries {2, 3,…, (n-2)} appear only once in the second row. In general, entries {1, 2, …, u} appear twice while entries {(u+1), …, (n-u-1)} appear only once in k$^{th}$ row of $N(K_n)$, where u = min{(k-1), (n-k)}, and k = 1, 2, …, (n–1).

In order to form graceful trees we need to select entries {1, 2, 3, …, (n–1)}, such that one entry gets selected from each row of $N(K_n)$, so that the corresponding pairs of vertices in $P(K_n)$ together form the graceful spanning tree of $K_n$.

The following are the number of choices for selecting the numbers from rows of $N(K_n)$ when n is **odd:**

| Number | n–1 | n –2 | n –3 | … | $\alpha+1$ | $\alpha$ | … | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|
| Choices | 1 | 2 | 3 | … | $\alpha$ | $\alpha$ | … | 2 | 1 |

Similarly, the number of choices for selecting the numbers from rows of $N(K_n)$ when n is **even** is given by

| Number | n–1 | n –2 | n –3 | … | $\alpha+1$ | $\alpha$ | … | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|
| Choices | 1 | 2 | 3 | … | $\alpha-1$ | $\alpha$ | … | 2 | 1 |

During the selection the first ($\alpha$–1) numbers {1, 2, …, $\alpha$–1} one has **at most two** choices in the rows from which they are selected and for rest of the numbers one has only one choice.

□

**Remark 3.1:** From this theorem it is clear that there are at most

$$C = \left[\frac{n}{2}\right]!\left[\frac{n}{2}-1\right]!2^{\left[\frac{n}{2}\right]-1} \text{ graceful trees.}$$

**Example:** Let $G \cong K_6$. We have

$$A(K_6) = \begin{bmatrix} 011111 \\ 101111 \\ 110111 \\ 111011 \\ 111101 \end{bmatrix}$$

$$P(K_6) = \begin{bmatrix} 0(1,2)(1,3)(1,4)(1,5)(1,6) \\ (2,1)0(2,3)(2,4)(2,5)(2,6) \\ (3,1)(3,2)0(3,4)(3,5)(3,6) \\ (4,1)(4,2)(4,3)0(4,5)(4,6) \\ (5,1)(5,2)(5,3)(5,4)0(5,6) \end{bmatrix}$$

$$N(K_6) = \begin{bmatrix} 0(1)(2)(3)(4)(5) \\ (1)0(1)(2)(3)(4) \\ (2)(1)0(1)(2)(3) \\ (3)(2)(1)0(1)(2) \\ (4)(3)(2)(1)0(1) \end{bmatrix}$$

In order to form graceful spanning trees (or, disconnected (n, n─1) graphs) we have to form nonzero monomials in $A(K_6)$ and the corresponding column content vectors with vertex pairs as elements in $P(K_6)$ in such a way that the corresponding elements in $N(K_6)$ together form the set of elements $\{1, 2, 3, 4, 5\}$.

Note that entry '5' is present only in the first row of $N(K_6)$ therefore the edge corresponding to vertex pair (1,6) in $P(K_6)$ must be present in every graceful spanning tree of $K_6$. Similarly, vertex pair (2,6) or (1,5) must belong to every spanning tree of $K_6$, and so on.

| Proper choice vectors from $N(K_6)$ | Corresponding vertex-pair vectors from $P(K_6)$ |
|---|---|

| | |
|---|---|
| (5, 1, 2, 3, 4) | ((1,6),(2,1),(3,1),(4,1),(5,1)) |
| | ((1,6),(2,1),(3,5),(4,1),(5,1)) |
| | ((1,6),(2,3),(3,1),(4,1),(5,1)) |
| | ((1,6),(2,3),(3,5),(4,1),(5,1)) |
| (5, 1, 3, 2, 4) | ((1,6),(2,1),(3,6),(4,2),(5,1)) |
| | ((1,6),(2,1),(3,6),(4,6),(5,1)) |
| | ((1,6),(2,3),(3,6),(4,2),(5,1)) |
| | ((1,6),(2,3),(3,6),(4,6),(5,1)) |
| (5, 2, 1, 3, 4) | ((1,6),(2,4),(3,2),(4,1),(5,1)) |
| | ((1,6),(2,4),(3,4),(4,1),(5,1)) |
| (5, 2, 3, 1, 4) | ((1,6),(2,4),(3,6),(4,3),(5,1)) |
| | ((1,6),(2,4),(3,6),(4,5),(5,1)) |
| (5, 3, 1, 2, 4) | ((1,6),(2,5),(3,2),(4,2),(5,1)) |
| | ((1,6),(2,5),(3,2),(4,6),(5,1)) |
| | ((1,6),(2,5),(3,4),(4,2),(5,1)) |
| | ((1,6),(2,5),(3,4),(4,6),(5,1)) |
| (5, 3, 2, 1, 4) | ((1,6),(2,5),(3,1),(4,3),(5,1)) |
| | ((1,6),(2,5),(3,1),(4,5),(5,1)) |
| | ((1,6),(2,5),(3,5),(4,3),(5,1)) |
| | ((1,6),(2,5),(3,5),(4,5),(5,1)) |
| (5, 4, 1, 2, 3) | ((1,6),(2,6),(3,2),(4,2),(5,2)) |
| | ((1,6),(2,6),(3,2),(4,6),(5,2)) |
| | ((1,6),(2,6),(3,4),(4,2),(5,2)) |
| | ((1,6),(2,6),(3,4),(4,6),(5,2)) |
| (5, 4, 1, 3, 2) | ((1,6),(2,6),(3,2),(4,1),(5,3)) |
| | ((1,6),(2,6),(3,4),(4,1),(5,3)) |
| (5, 4, 2, 1, 3) | ((1,6),(2,6),(3,1),(4,3),(5,2)) |
| | ((1,6),(2,6),(3,1),(4,5),(5,2)) |
| | ((1,6),(2,6),(3,5),(4,3),(5,2)) |
| | ((1,6),(2,6),(3,5),(4,5),(5,2)) |
| (5, 4, 2, 3, 1) | ((1,6),(2,6),(3,1),(4,1),(5,4)) |
| | ((1,6),(2,6),(3,1),(4,1),(5,6)) |
| | ((1,6),(2,6),(3,5),(4,1),(5,4)) |
| | ((1,6),(2,6),(3,5),(4,1),(5,6)) |
| (5, 4, 3, 2, 1) | ((1,6),(2,6),(3,6),(4,2),(5,4)) |
| | ((1,6),(2,6),(3,6),(4,2),(5,6)) |
| | ((1,6),(2,6),(3,6),(4,6),(5,4)) |
| | ((1,6),(2,6),(3,6),(4,6),(5,6)) |
| (5, 4, 3, 1, 2) | ((1,6),(2,6),(3,6),(4,3),(5,3)) |

| | |
|---|---|
| | ((1,6),(2,6),(3,6),(4,5),(5,3)) |

We have thus forty graceful spanning trees that can exist in $K_6$. As per the lemma 3.1 $K_6$ can have at most forty-eight graceful spanning trees.

It is known that there are in all 6 (unlabeled) spanning trees up to isomorphism in $K_6$, and one can easily check that graceful representative for every nonisomorphic tree in more than one copies is present in the list given above.

Is graceful labeling unique ? No. the following simple theorem provides the easiest other graceful labeling:

**Theorem 3.3** Every graceful (n, n─1) tree remains graceful under the transformation (mapping) of vertex labels:
$j \to (n-j+1)$.

**Proof:** Let $i, k$ be the vertex labels of two adjacent vertices of the tree. Then the edge label for this edge will be $|i-k|$. Now under the mentioned transformation the edge labels
$|i-k| \to |(n-i+1)-(n-k+1)| = |i-k|$, hence etc.

□

**4. Generation of (n, n─1)-trees from (n─1, n−2)-trees:** We now give some results for generating all possible unlabeled (i.e. nonisomorphic) (n, n─1) trees from all possible unlabeled (nonisomorphic) (n─1, n-2) trees by extension as described below:

**Definition 4.1:** Let *G* be an unlabelled (p, q) graph and let $G^e$ be a supergraph of *G* obtained by taking a (new) vertex outside of the vertex set *V(G)* and joining it to some (unspecified) vertex of *G* by an (new) edge not in the edge set of G, E*(G)*, is called the **extension** of *G* to $G^e$.

**Definition 4.2:** The subset $V_j$ of vertices $\{u_1{}^j, u_2{}^j, \cdots, u_r{}^j\}$ in a tree T is called a **set of equivalent vertices** or simply an **equivalent set** if all the trees T + $vu_s{}^j$, 1 ≤ s ≤ r, obtained from T by adding an edge $vu_s{}^j$, obtained by joining vertex $u_s{}^j$ in set $V_j$ to a new vertex $u$ not in *V(G)*, are isomorphic.

**Definition 4.3:** The subset of vertices (vertices) of *V(G)* is called a **set of equivalent vertices** or simply an **equivalent set** if the extension of graph *G* at any vertex among these vertices, achieved by joining any vertex among these vertices to a (new) vertex outside, not in *V(G)*, leads to graphs which are all isomorphic.

**Definition 4.4:** The collection of subsets $\{V_1, V_2, \cdots, V_m\}$ of *V(G)*, the vertex set of graph *G* is called a **partitioning of *V(G)* into equivalent sets** if all the subsets $V_i$ , i = 1, 2,…, m are equivalent sets,

$$V_i \bigcap V_j = \phi, \forall\, i \neq j, and\ V(G) = \bigcup_{i=1}^{m} V_i$$ , where $\phi$ is a null set.

**Definition 4.5:** The collection of all possible unlabelled (nonisomorphic) (n, n—1) trees is called **(n, n—1)-stock**.

**Definition 4.6:** The set of unlabelled (nonisomorphic) (n+1, n) trees obtained by extension at (any) one vertex belonging to every set of equivalent vertices in the partitioning of *V(T)* into equivalent sets for a tree *T* is called **Complete Extension of *T*,** and is denoted by ***CE(T).***

**Theorem 4.1:** The collection of nonisomorphic trees contained in $\bigcup_j {}^{/} CE(T_j)$, $T_j$ belongs to (n,n—1)-stock, and $^{/}$ indicates that the union is over nonisomorphic trees, forms a (n+1,n)-stock.

**Proof:** For every tree *T* belonging to (n+1, n)-stock there exists trees *T** of some isomorphism type in (n, n—1)-stock obtained by deleting pendant vertex of *T*. So, *T* can be considered as arrived at by extension of some tree like *T** belonging to (n, n—1)-stock, and trees isomorphic to T arriving from more than one *T** is taken only once in the union.

□

We now proceed to obtain (n+1, n)-stock of trees from (n, n-1)-stock in a different way. We proceed to show that it is possible to achieve this task by extending at some specially chosen **special vertices** of the trees in the (n, n-1)-stock. We first proceed to define these special vertices and then proceed to show that it is possible to build the entire (n+1, n)-stock by extension at these special vertices of the trees belonging to (n, n-1)-stock.

Let us denote by $p(T)$ the set of pendant vertices of a tree $T$ .

**Definition 4.6:** A vertex in a tree is called **prependant** if when all the pendant vertices adjacent to it are deleted, then the vertex itself becomes a pendant vertex.

We denote the set of prependant vertices in a tree $T$ by $pr(T)$. Thus, if a vertex $u \in V(T)$ is prependant then we say (or denote this by) $u \in pr(T)$.

**Definition 4.7:** A vertex in a tree which is itself a pendant vertex and further adjacent to a prependant vertex then such vertex is called **pendant vertex adjacent to a prependant vertex**.

We denote the set of pendant vertices adjacent to prependant vertices in a tree $T$ by $ppr(T)$. Thus, if a vertex $u \in V(T)$ is pendant vertex adjacent to a prependant vertex then we say (or denote this by) $u \in ppr(T)$.

**Lemma 4.1:** For n > 2, every (n, n−1)-tree contains a prependant vertex.

**Proof:** It is clear to see that when we delete a pendant vertex of any tree the graph left behind is a again a tree. Now, Suppose every vertex adjacent to pendant vertex is not prependant then if all the pendant vertices are deleted the tree that will result will not have any pendant vertex, a contradiction since it is well-known that every tree must contain pendant vertices (at least two), hence etc.

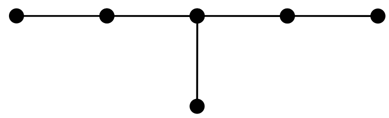
□

**Remark 4.1:** Thus every tree contains at least two special vertices, prependant vertex and pendant vertex adjacent to a prependant vertex.

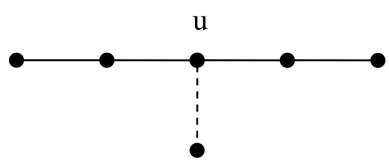
**Definition 4.10:** A vertex in a tree is called a **special vertex** if it belongs to sets $pr(T)$ or $ppr(T)$.

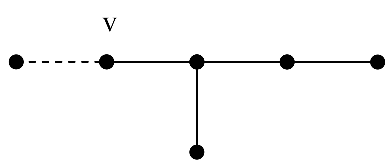
**Remark 4.2:** It is important to note that a tree in the higher stock can be obtained by extension at more than one special vertex and these are special vertices of distinct nonisomorphic trees in the stocks lower by one vertex in size. On the other hand, there are some trees which can be obtained by extension at a vertex of some unique tree in the lower stock.

**Examples:** Consider following tree:



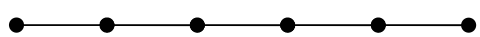
This tree can be obtained by extension at certain vertices of the following distinct nonisomorphic trees in the lower stock and the extension at these vertices of two distinct trees is shown by the dotted edge.



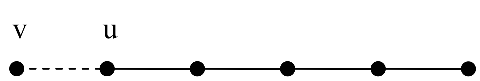




Here, vertex with label “u” is not special for the extended (higher ordered) tree but vertex “v” of the extended tree is special (prependant vertex).

Consider the following tree:



This tree can be obtained from the following unique tree in the lower stock by extension. The extension at the vertex “u” is shown by the dotted edge. Here vertex with label “u” is special (prependant vertex). Also the vertex with label “v” is special (pendant vertex adjacent to a prependant vertex).




**5. Graceful Tree Conjecture (First Approach):** In this section we settle the graceful tree conjecture using our first approach as mentioned in the abstract, in the affirmative by induction. Our aim in this section is to prove the result that every tree can be gracefully labeled by starting with assigning label 1 to any chosen special vertex and by further assigning other suitable labels 2, 3, 4, … etc for other vertices of the tree.

We consider now few examples of trees and check how we can have graceful labeling to them by assigning label 1 to some special vertex and choosing further other suitable labels form numbers 2, 3, 4, …., etc. We choose a prependant vertex among the special vertices to assign label 1. The same can be done by choosing a pendant vertex adjacent to prependant vertex. This can be readily seen by assigning complementary labeling for this gracefully labeled tree by using theorem 3.3.

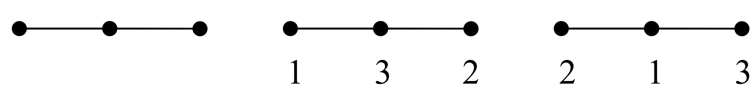
**Theorem 5.1:** It is possible to gracefully label any tree by assigning smallest label (= 1) to any one of its special vertex.

**Proof:** We proceed by induction on n, the number of vertices in the tree.
**Step 1:** Let n = 1, 2, 3, 4.

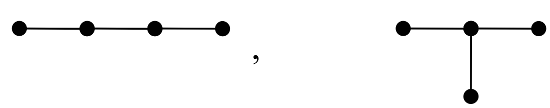
(a) When n = 1, we have a vertex tree or (1,0)-tree where the vertex can be thought of as special and we assign label 1 to it which makes it graceful.

(b) When n = 2, we get a (2,1)-tree with single edge and its both vertices special. We can assign label 1 to (any) of these special vertices and label 2 to other vertex which is a graceful labeling.

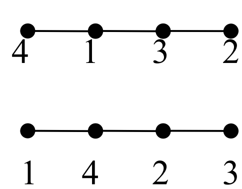
(c) When n = 3, we get the following only unlabeled tree up to isomorphism and both its pendant vertices as well as middle vertex are special and we have the following two gracefully labeled trees, such that label 1 is assigned to a special vertex in the both the cases as desired and as shown below:

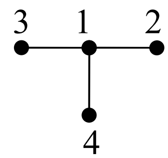


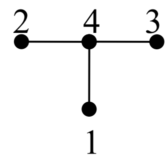

(d) When n = 4, we have up to isomorphism the following two trees:



As shown below we have gracefully labeled trees for both trees and label 1 can be assigned to special vertices as shown in the below figures:






**Step 2:** By induction we can assume that the existence of graceful labeling to every (k, k-1)-tree up to isomorphism with label 1 assigned special vertices. So, we assume the result by induction for n = k > 4 and proceed to prove it for n = (k+1).

Take any (k+1, k)-tree which is unlabelled. Consider set of its special vertices.

1) Consider among special vertices, a vertex which is also a prependant one. Suppose this prependant vertex is such that when the pendant vertex adjacent to it is deleted then either the prependant vertex still remains prependant or becomes now a pendant vertex which is further adjacent to a prependant vertex, i.e. the prependant vertex still remains a special vertex. In this case the result will easily follow. Because we can label this special vertex, which either has remained prependant vertex or has become a pendant vertex adjacent to prependant vertex, with label 1 and obtain graceful labeling for this edge deleted tree by induction. Further, we reattach the deleted edge and assign label (k+1) to end vertex of the attached edge and thus we have graceful labeling for (k+1, k)-tree under consideration, and thus **we are done**.

2) Suppose the tree under consideration doesn't contain a prependant vertex such that when the pendant vertex adjacent to it is deleted this prependant vertex still remains prependant or becomes now a pendant vertex which is adjacent to a prependant vertex, i.e. this vertex still remains a special vertex. This situation, where there doesn't exist a vertex which is originally prependant but remains special after deletion of pendant vertex adjacent to it, requires a careful detailed consideration. In this situation we consider the following four cases:

**Case 1:** In this case there is only one pendant vertex adjacent to a prependant vertex under consideration and the degree of such prependant vertex is two.

**Case 2:** In this case there are “t” pendant vertices adjacent to a prependant vertex under consideration and the degree of such prependant vertex is (t+1).

**Case 3:** In this case there is only one pendant vertex adjacent to a prependant vertex under consideration and the degree of such prependant vertex is strictly more than two.

**Case 4:** In this case there are “t” pendant vertices adjacent to a prependant vertex under consideration and the degree of such prependant vertex is strictly more than (t+1).

Out of these four cases Case 3 and Case 4 are redundant as they contradict with the definition of a prependant vertex and so do not exist. So, we need only to take into consideration Case 1 and Case 2. We begin our discussion with Case 1.

For this Case 1 we carry out the following steps:

a) We choose a prependant vertex, v say, in the given (k+1, k)-tree, T say.
b) We delete some pendant vertex other than the one adjacent to v, u say, where u is away from the pendant vertex adjacent to v to obtain tree T-u. In other words, the pendant vertex chosen to be deleted does not belong to the star type subtree formed around vertex with label v.

**Remark 5.1:** It can be assumed that such a pendant vertex exists because otherwise tree T will be a star tree and it is well known that star tree can be gracefully labeled.

c) With b) the tree T-u has now become (k, k-1)-tree. So, induction applies. So, now we gracefully label this tree by choosing label 1 for the chosen prependant vertex v in a), label k for the pendant vertex adjacent to this prependant vertex and suitable label (k-1) for other adjacent vertex to this prependant vertex, label 2 for the vertex adjacent to vertex labeled (k-1), and other suitable labels for other vertices such as (k-2), (k-3), ….., etc among the totality of labels {1, 2, 3, …., k}. As a result we have a gracefully labeled copy of tree T-u.
d) We now carry out the following changes in the labeling of this gracefully labeled copy of tree T-u: We change label “(k-1)” to label “k” and label “k” to label “(k+1)”. So, we have now tree T-u

labeled with labels {1, 2, (k-2), k, (k+1)}. Thus, label (k-1) will now be absent in this new labeling of tree T-u.

**Remark 5.2:** Note that since initially, as per c), (k, k-1)-tree T-u was gracefully labeled. So, it has all separations among {1, 2 , …, j, …, (k-2), (k-1)} present, obtained through finding the differences of vertex labels of adjacent vertices, in it. Now, after performing changes in labels as per d), the following changes take place. Separations taken with vertex labeled as 1 change as follows: Separations (k-1) → k, (k-2) → (k-1). Due to change in label "(k-1)" to label "k" the old separations with respect to vertex having old label (k-1), say {d1, d2, d3, …,} etc. When now taken with respect to vertex newly labeled as k the old separations will now change to new separations, namely, {d1+1, d2+1, d3+1, …} etc. Note that all other (old) separations will remain intact and will be the same as before. Thus, the original set of separations {1, 2, 3, … , (k-1)} will now change to {1, 2, …. , k} with some one difference $j, 1 \le j \le k$ absent in this set of separations. Further, all the set of new labels will be {1, 2, …, k-2, k, k+1} and only label (k-1) is missing.

e) We now reattach the deleted pendant vertex and its corresponding deleted edge at the same place and to the same vertex (which was labeled "u" and has now become during graceful labeling "w" say) and assign label "(k-1)" instead of original label "v" to it since "(k-1)" is now the only missing vertex label. Now, if the separation |w-(k-1)| = *j* then **we are done** because the original (k+1, k)-tree obtained in d) was only short of missing separation *j* from becoming gracefully labeled.
f) Suppose now that |w-(k-1)| is not equal to *j* and further suppose that |w-(k-1)| = d. In this case, it is clear to see that though we have done labeling of the (k+1, k)-tree with distinct labels {1, 2, …, k+1} for tree T still it is not graceful as it has all separations distinct except exactly one separation. Thus, its corresponding set of separations will be {1, 2, …., d, d, ….., k} which contains all required separations except separation "*j*" and so has exactly one separation "d" with "double" occurrence, where, d ∈ {1, 2, …, k}.

We now proceed with the following simple result:

**Lemma 5.1:** Let T be a gracefully labeled tree and V(T) = {1, 2, …, n}, where V(T) represents the set of labels for vertices of T. Now if we transform these vertex labels *j* → (*j* + m), where m is some integer, for all vertices with vertex labels *j* in V(T) then T still remains graceful.

**Proof of the lemma:** This is a straightforward result because with this transformation the original separations remain invariant. This is so because due to this transformation the original labels of some two adjacent vertices, say u, v in V(T) now change to (m + u), (m + v) respectively and the original and changed separations remain equal. i.e. $|u - v| = |(m + u) - (m + v)|$.

□

g) Now consider the tree obtained after doing actions depicted in d) and e). Consider further the subtree of this tree that exists other than the path formed by edges connecting vertex with label (k+1) to vertex with label 1, further connecting vertex with label 1 to vertex with label k, and further connecting vertex with label k to vertex with label 2. This subtree will contain vertices with labels {2, …, (k-1)} and the corresponding difference set, obtained by taking and collecting differences of adjacent vertices, will be {1, 2, …, (k-2)} with some difference missing and some other difference occurring twice. Now, induction applies and taking lemma 5.1 into consideration we can make this subtree graceful. In other words, there must exist some permutation which when applied on the set of vertex labels {2, …, (k-1)} should cause the elimination of the only difference occurring twice and should make the subtree graceful. Now if we consider again the entire tree we have differences (k-1) and k become available and the missing vertex labels (k+1), 1, k are also become available and thus **we are done** with Case 1.

We begin our discussion with Case 2. This case is easy because when there are more than one pendant vertices adjacent to the prependant vertex under consideration this prependant vertex remains prependant (and so special) after deleting any one of its pendant vertex. So, after deleting such pendant vertex from (k+1, k)-tree T we are left with (k, k-1)-tree and choosing label 1 to the vertex that remained prependant we can gracefully label this tree by induction and reattach the deleted vertex and label it by label (k+1) and thus we have gracefully labeled tree T. So again **we are done** for this only left out Case 2.

□

**6. Graceful Tree Conjecture (Second Approach):** In this section we proceed with an altogether different approach, our second approach, as mentioned in the abstract. We show that all graceful trees in a complete graph are essentially certain paths, called lattice paths, in a triangular shaped lattice of points.

**Definition 6.1:** A **delta lattice (n-delta lattice)** is a triangular shaped lattice of points, having shape of an inverted triangle, such that certain distinct vertex pair (edge) has been associated as a label with each point of this lattice, and the vertex pairs (each representing a unique edge) are assigned to these lattice points in such a way that the lattice points in the top row have the associated labels (i, i+1), where i goes from 1 to n-1, the lattice points in the second row below the first row have the associated labels (i, i+2), where i goes from 1 to n-2, …., the lattice points in the k-th row, reached by successively creating rows downwards, have the associated labels (i, i+k), where i goes from 1 to n-k, … the last row has a single lattice point with vertex pair (1, n) as the associated label.

We give below as an illustration the representation of this lattice with associated labels for n = 2, 5, 6 **(we don't draw here the associated lattice points and it is to be understood that they are there)** as follows:

1) For n = 2, the 2-delta lattice consists of single lattice point labeled by the associated vertex pair (1,2) :

(1,2)

2) For n = 5, the 5-delta lattice is:

(1,2)  (2,3)  (3,4)  (4,5)

(1,3)  (2,4)  (3,5)

(1,4)  (2,5)

(1,5)

5) For n = 6, the 6-delta lattice is:

(1,2)     (2,3)     (3,4)     (4,5)     (5,6)

(1,3)     (2,4)     (3,5)     (4,6)

(1,4)     (2,5)     (3,6)

(1,5)     (2,6)

(1,6)

**Remark 6.1:** It is easy to visualize that n-delta lattice is essentially a representation for complete graph on n vertices where these vertices are labeled by numbers {1, 2, 3, …, n}.

**Definition 6.2:** An imaginary vertical line starting from lattice point associated with pair (1,n) and going upwards passing through the lattice points with labels (2,n-2), (3,n-3), ….., extending and incorporating the lattice points on the rows, and rising up to first row is called **line of symmetry**.

**Definition 6.3:** The **off line axis** is an imaginary line passing through the lattice points having associated labels in terms of the vertex pairs {(1,n), (2,n),(3,n),….,(n-1,n)}.

**Definition 6.4:** A **lattice path** is a path obtained by selecting some one lattice point on each row of n-delta lattice and joining these lattice points in sequence starting with the lattice point on the lowest row and moving up in succession incorporating the chosen lattice point on each row till the path finally terminates at the selected lattice point on the first row.

**Definition 6.5:** A **piece** of a lattice path is certain portion of lattice path, may be consists of single lattice point with label (i, j) that represents an edge with end vertices having labels i and j, or certain portion of lattice path obtained by joining some lattice points in successive rows of n-delta lattice, or the entire lattice path itself.

**Definition 6.6:** Let (x1, y1)→ (x2, y2)→…. →(xk, yk) be a piece of lattice path. The totality of distinct vertices present in the pairs (xi, yi), i = 1, 2, …., k, is called the **vertex cardinality of piece**, **and it is denoted by v.**

**Definition 6.7:** Let (x1, y1)→ (x2, y2)→…. →(xk, yk) be a piece of lattice path. The totality of distinct edges present (k in number in the present case) is called **edge cardinality of piece**, **and is denoted by e**.

In the above illustrations of delta lattices:

1) For n = 2 the line of symmetry passes through lattice point associated with vertex pair (1,2), i.e. through the only lattice point.
2) For n = 5 the line of symmetry passes through lattice point associated with vertex pairs (1,5), (2,4) since there is no lattice point on this vertical line from first and third row.
3) For n = 6 the line of symmetry passes through lattice point associated with vertex pairs (1,6), (2,5), (3,4) since there is no lattice point on this vertical line from second and fourth row.

**Remark 6.2:** In the above definition by starting with the selected lattice point on the first row and moving down in succession incorporating the chosen lattice point on each row till the path finally terminates at the selected lattice point on the last row we will construct the same lattice path.

**Remark 6.3:** It is easy to visualize that a lattice path in n-delta lattice, when corresponds to a tree, is essentially equivalent to showing existence of a gracefully labeled isomorphic copy (for an unlabeled tree of some isomorphic type) in the complete graph on n vertices where these vertices are labeled by numbers {1, 2, 3, …, n}.

We now proceed to give as examples some lattice paths in the above mentioned lattices for n = 2, 5, 6. The examples of right lattice paths can be constructed in similar way.

1) Case n = 2: In this case, there is only one lattice point with associated vertex label (1,2). So, the lattice path is of unit length.
2) Case n = 5: Some lattice paths in this case are as follows:

   (1,5)→(1,4)→ (1,3)→(1,2)

   (1,5)→(1,4)→ (1,3)→ (2,3)

   (1,5)→(1,4)→ (2,4)→ (2,3)

3) Case n = 6. Some lattice paths are as follows: The first three can be considered as obtained by extending the paths for case n = 5 by appending with (1,6).

(1,6)→(1,5)→(1,4)→ (1,3)→(1,2) .....(1)

(1,6)→(1,5)→(1,4)→ (1,3)→ (2,3) .....(2)

(1,6)→(1,5)→(1,4)→ (2,4)→ (2,3) .....(3)

the other four are new, arising due to incorporation of new lattice points with associated vertex pairs (1,6), (2,5) and (3,4) as labels for these new lattice points while we move from the 5-delta lattice to the 6-delta lattice (which were not existing in the 5-delta lattice) and these new lattice paths are

(1,6)→(1,5)→(2,5)→(2,4)→(2,3) .....(4)

(1,6)→(1,5)→(1,4)→(2,4)→(3,4) .....(5)

(1,6)→(1,5)→(2,5)→(2,4)→(3,4) .....(6)

(1,6)→(1,5)→(2,5)→(1,3)→(3,4) .....(7)

It is easy to see that when we take a lattice path and use it to construct a graph by taking the vertex pairs that appear in that path as edges and the numbers that appear in the totality in these vertex pairs as vertex labels we get essentially a graceful graph.

If this graceful graph is an (n,n-1) connected graph or (n,n-1) acyclic graph then the lattice path represents a graceful tree. Otherwise, the associated graph, though graceful, obtained from that lattice path is not a tree.

Further, it is easy to see that if we take some lattice path and consider the path formed as mirror image in the line of symmetry of the chosen path then the trees (or not trees) associated with these lattice paths are isomorphic. This is clear from the fact that as the mirror image of a lattice point with associated vertex pair (i, j) we get the lattice point with associated vertex pair (n-i+1, n-j+1) and so by theorem 3.3 proved above the result follows.

In the case of 6-delta lattice we have considered here some seven lattice paths (as given above). It can be easily seen that all these lattice paths represent graceful trees. Trees associated with the five lattice paths numbered by (1), (2), (5), (6), (7) are non-isomorphic, while trees associated with the lattice paths numbered by (3), (4) constitute the pair of isomorphic trees. So, these lattice paths form graceful labeling for all the six non-isomorphic trees that exist on six vertices.

Consider following two **straight** lattice paths which are symmetrically placed (mirror images of each other) around line of symmetry, namely,

$$(1,n)\rightarrow(1,n-1)\rightarrow(1,n-2)\rightarrow\dots\dots\dots\rightarrow(1,2)$$

and

$$(1,n)\rightarrow(2,n)\rightarrow(3,n)\rightarrow\dots\dots\dots\rightarrow(n-1,n)$$

It is easy to check that these lattice paths lying at left and right boundary of n-delta lattice correspond as a graph to gracefully labeled (n,n-1) star trees.

Consider following two **zigzag** lattice paths which are symmetrically placed (mirror images of each other) around line of symmetry, namely,

$$(1,n)\rightarrow(1,n-1)\rightarrow(2,n-1)\rightarrow(2,n-2)\rightarrow(3,n-2)\rightarrow\dots\dots$$

and

$$(1,n)\rightarrow(2,n)\rightarrow(2,n-1)\rightarrow(3,n-1)\rightarrow(3,n-2)\rightarrow(4,n-2)\rightarrow\dots$$

It is easy to check that these lattice paths going away from and coming towards line of symmetry by unit distance at each alternate move and passing in a zigzag way close to line of symmetry of n-delta lattice correspond as a graph to gracefully labeled (n,n-1) path trees.

We now proceed with an algorithm to generate all possible gracefully labeled trees in terms of the totality of all lattice paths in n-delta lattice.

**Algorithm 6.1:**

1) Take n-delta lattice. Select lattice point (1, n). We have formed a tree with single edge.
2) Extend lattice path to be formed by appending to chosen pair (1, n) some pair (i, j) such that exactly one new vertex gets added in the number of totality of distinct vertices that are present at the time of appending this new pair and continue, by going up to some row or coming down to some row for selection of next lattice point (and the corresponding associated pair) for appending during the lattice path formation process, till finally exactly one pair gets selected from each row of n-delta lattice. We have thus formed a complete

lattice path (which can be shown as to be arrived at by connecting chosen pairs by starting at the bottom most row and going up sequentially to next row upwards) representing gracefully labeled copy of some (n, n-1)-tree up to isomorphism.

□

For the sake of completeness let us state some definitions:

**Definition 6.8:** A tree is called a **star tree** or simply a **star** if it is a tree with one vertex of degree k, k bigger than one, and all other vertices are adjacent to it and have degree exactly equal to one.

**Definition 6.9:** The vertex of degree k, k bigger than one, in a star tree is called **centre** of this star tree.

**Definition 6.10:** A tree is called a **path tree** or simply a **path** if it is tree with all vertices have degree two except two (end) vertices (where the path terminates) and they have degree one.

**Definition 6.11:** While assigning labels to vertices in a tree we have assigned to some vertex a label, x say. After this assignment label, y say, is assigned to a vertex adjacent to x. The label y is said to be **farthest** from label x if $|x – y|$ is maximal among the available labels at that stage of labeling.

**Definition 6.12:** In a tree, formed by certain star trees and certain path trees adjacent to each other in certain predefined way, a star (or a path) in it is called **peripheral** if there exists at least one pendent vertex attached to (or, as a part of) that star or path.

**Definition 6.13:** A peripheral star (or a two path), as a part of some tree is called **special** if it is centered at a prependant vertex.

**Lemma 6.1:** Every tree contains, as a part, a peripheral star centered at a prependant vertex.

**Proof of the lemma:** Every tree contains a prependant vertex by lemma 4.1, and the star centered at this prependant vertex, as a part of the (big) tree under consideration, as a peripheral star centered around prependant vertex.

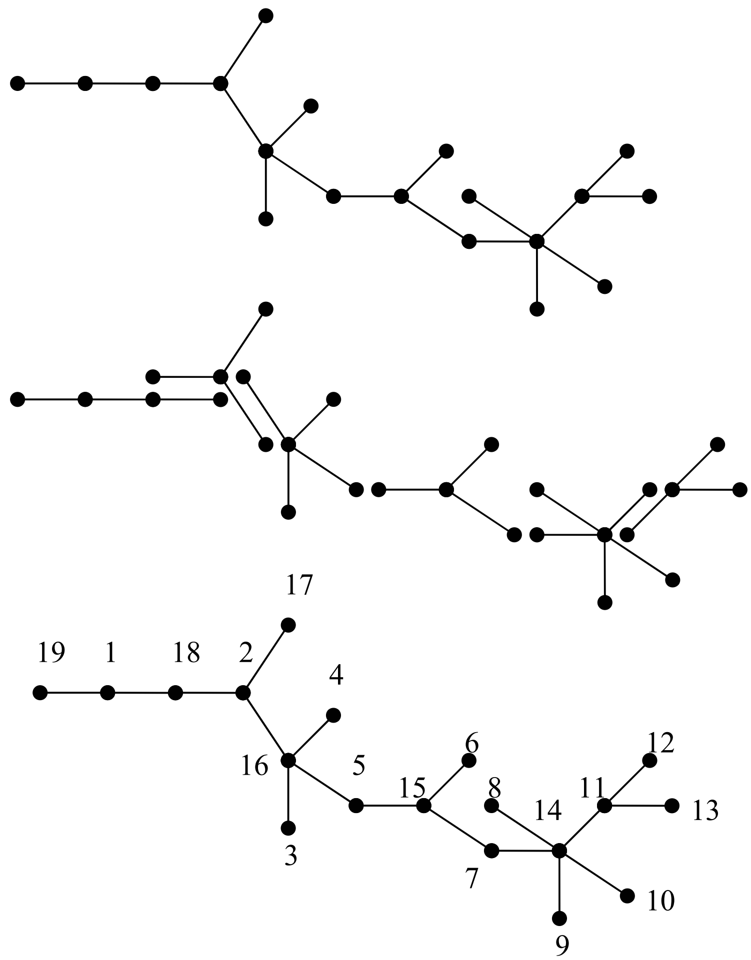
□

**Remark 6.4:** All other trees are made up of joining of certain star trees and certain path trees in certain predefined way such that they have some edges and/or vertices in common, i.e. it is possible to think about any tree as made of juxtaposition of certain stars and certain paths of various sizes in a predefined way such that they have some edges and/or vertices in common, to get the desired isomorphism type of the tree under consideration. By analyzing this well defined structure of a tree we can chalk out our way to proceed with the search for the associated lattice path in the n-delta lattice for a tree of order n by starting labeling **at some peripheral star centered at a prependant vertex** by selecting edges starting at the bottom of n-delta lattice. Note that as we move up starting at the bottom of delta lattice forming the desired lattice path the separation available for selection go on lowering with every selection. In order to keep easy availability of appropriate labels along with appropriate separations one may require to proceed with labeling, after assigning label 1 to some prependant vertex, so that among the adjacent vertices to vertex with label 1 the **farthest label** may be required be assigned to the adjacent vertex of **smallest degree**, and so on.

Let us now consider few simple examples:

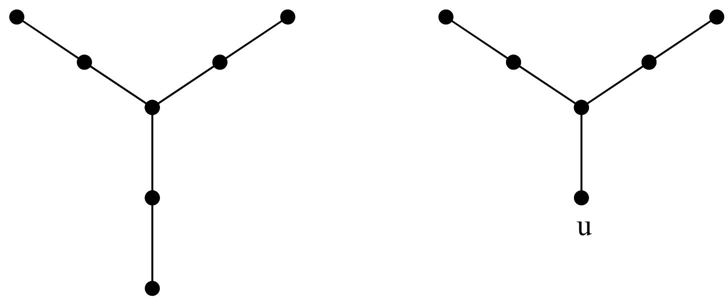
**Example 1:** Suppose we are given a tree made up of a star tree with p pendant vertices and a path tree of length q where the path emerges from one of the pendent vertices of the star tree under consideration, then the lattice path will be partly straight, starting at lattice point having associated vertex pair (1,n) as its label and then going upwards along any boundary line to incorporate (p-1) lattice points (i.e. till the star tree of order (p+1) is not formed in this process) and then it will continue further in the upwards direction in a zigzag way (till the path tree of order q is not formed in this process) in the n-delta lattice.

**Example 2:** In the first figure we show an unlabeled (19, 18)-tree. In the figure just below it we have shown how this tree can be broken into subtrees, either paths or stars, where nearby subtrees have some edges and /or vertices in common. We consider 19-delta lattice and by starting with leftmost subtree, which is a 3-path in this case, and also starting from bottom of this 19-delta lattice we select appropriately a zigzag path in this lattice for subtrees which are paths and a straight lattice path (by going diagonally or off-diagonally) for subtrees which are stars, and taking appropriate care for overlaps of vertices or edges. We thus finally determine **the lattice path** that represents the gracefully labeled version of given unlabeled tree as shown in the last figure.




**Definition 6.14:** A vertex in an (n, n-1) unlabeled tree in n-stock is called **essential** if we need to extend at this vertex by attaching a new edge connecting to a newly taken vertex to get an unlabeled tree in (n+1)-stock.

**Example:** In the figure shown below (to get tree shown at left in 7-stock) the vertex labeled “u” in tree shown to right in 6-stock is essential vertex.




**Remark 6.5:** It is **our belief** that **in the totality of lattice paths representing graceful trees** formed by joining lattice points obtained by starting with the selection of bottom most lattice point, associated with edge joining vertex labeled 1 and vertex labeled n and represented by pair (1, n), and choosing exactly one lattice point, and the associated pairs, to incorporate in the lattice path from each row of the n-delta lattice every tree up to isomorphism is contained in it. In actuality, there are in fact more than one lattice paths which are associated with a tree of same isomorphism type, i.e. there are more than one gracefully labeled versions of the same tree up to isomorphism. What is the important difference to be noted in these graceful copies of the same tree? Note that in these copies label 1 has changed its place in such a way that in these different copies label 1 gets itself associated with different vertices of the tree which are inequivalent (See definition 3.4) and in the process cover all vertices as a subset which are essential (See definition 6.14). Due to this we get all the trees which are gracefully labeled and lying in the next, i.e. (n+1)-stock by attaching edge (1, n+1) to the vertices labeled 1 in these trees, and so on.

**Definition 6.15:** A **lattice monomial** is a representation of lattice path of length k formed by connecting k pairs of vertices in terms of a monomial in k number of variables.
**Example:** With the lattice path (1,6)→(1,5)→(1,4)→ (1,3)→(1,2) we associate the lattice monomial $x_{16}x_{15}x_{14}x_{13}x_{12}$ . By lexicographic ordering this monomial can be written as $x_{12}x_{13}x_{14}x_{15}x_{16}$ .

**Remark 6.6:** These monomials in double indexed variables can easily be lexicographically ordered by first ordering the variables with respect to first index and when first index is equal then with respect to second index. For example, for two double indexed variables $x_{ab}, x_{cd}$ we have

$x_{ab} \leq x_{cd}$ iff $a < c$ or ( $a = c$ and $b \leq d$ ). Note that lattice paths can be ordered by defining lexicographic order on the associated lattice monomials.

**Remark 6.7:** It is well known that isomorphism of two rooted trees can be decided in linear time. There are well known algorithms for testing isomorphism of rooted and unrooted (n, n-1)-trees with order $O(n)$ and $O(n^2)$ respectively. Different rooted or unrooted trees can be distinguished by introducing a total order associated with them. For example, there are easy algorithms described in [5] to determine the isomorphism of rooted and unrooted trees with order $O(n)$ and $O(n^2)$ respectively. When two rooted trees are isomorphic they are also isomorphic as unrooted trees under the same isomorphism map.

**Remark 6.8:** For every unlabeled tree in n-stock there are sufficiently many lattice paths associated with it in n-delta lattice so that every vertex which is essential in nature in this tree gets label 1 through some graceful representation in terms of some lattice path representing this unlabeled tree. Thus, all lattice paths representing trees in the n-delta lattice contain all trees up to isomorphism in the n-stock and further in sufficiently many number to have label 1 to all essential vertices to generate lattice paths, to be obtained by extending at label 1 by new edge represented by new lattice point associated with pair (1, n+1) connecting new vertex labeled (n+1) taken newly, representing trees in the (n+1)-delta lattice to contain all trees up to isomorphism in the (n+1)-stock by extension by appending lattice point corresponding to pair (1, n+1).

**Definition 6.15: Leveled representation** of a rooted tree, called **leveled rooted tree,** or **rooted tree with levels,** is the one in which we associate level "0" with root. Level "1" contains vertices adjacent to root. Level "2" contains vertices adjacent to vertices in level "1", and so on.

**Remark 6.9:** If a vertex belongs to level "k" then root is connected to it by a path of length (k-1), or, if the vertex is at distance (k-1) from the root.

**Remark 6.10:** Let T be a given rooted leveled tree. We associate a number $\alpha$ with each vertex in the level "k" representing the number of 1-paths emerging from it and terminating at distinct vertices in "(k+1)-th" level, i.e. $\alpha$ associated with each vertex in the level "k" is count of vertices that belong to "(k+1)-th" level and adjacent to it. We order the numbers, $\alpha^{s}$, associated with all the vertices present in every level, k, in lexicographically non-increasing order and create a single number, say $\beta_k$ . We further create **one more number** by writing these numbers $\beta_k$ in a sequence, **say** $\lambda$ , and, $\lambda = \beta_0\beta_1\beta_2\cdots\beta_n$

**Definition 6.15:** The **length** of number $\beta_k$ is the count of numbers present in $\beta_k$ irrespective of their values.

**Remark 6.11:** It is easy to check that the sum of $\alpha$ $^{s}$, associated with all the vertices present in level k length of level (k+1). Also, number of vertices present in tree T represented by corresponding $\lambda$ is equal to sum of lengths of all levels, and further this is equal to sum of values of numbers present in all levels plus one (as root is not counted in this sum).

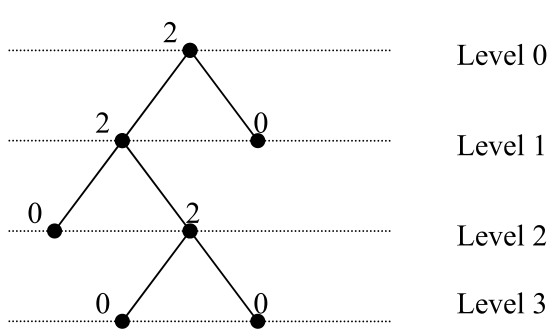
Here is (refer to definition 6.15 and remarks 6.9, 6.10) **one more** simple characterization of rooted trees that works in linear time:

**Lemma 6.2:** With every rooted leveled tree there is associated a unique number $\lambda$ that characterizes it up to isomorphism, i.e. for isomorphic rooted trees the associated $\lambda$ -number is identical.

**Proof:** When rooted trees T and T* are isomorphic one can overlap (put one on the other) these trees using isomorphism map and the rest is clear. Now, suppose $\lambda$ -numbers associated with trees T and T* are identical. In this case, we can construct T and T* in identical steps and visualize the identity of trees at each stage of there construction.

□

**Illustrative Example:** We illustrate the concepts stated onwards from definition 6.15 with the help of following tree, T say:




In the above example we have:

$\beta_0 = 2$, $\beta_1 = 20$, $\beta_2 = 20$, and $\beta_3 = 00$ and therefore,
$\lambda = 2\ 20\ 20\ 00$. Note that $\lambda$ characterizes this rooted tree T up to isomorphism.
L($\beta_0$) = 1, L($\beta_1$) = 2, L($\beta_2$) = 2, L($\beta_3$) = 2, where "L" denotes the length.
SV($\beta_0$) = 2, SV($\beta_1$) = 2, SV($\beta_2$) = 2, SV($\beta_3$) = 0, where "SV" denotes the sum of values of numbers occurring at respective levels.
It is clear to see that Sum(SV($\beta_k$)) = L($\beta_{k+1}$), and cardinality of vertex set of tree T is Sum(L($\beta_k$)) = Sum(SV($\beta_k$)) + 1.

**The $\lambda$ representation of a rooted tree enables us to understand how the tree is structured**, i.e. it is made up of which sub-trees juxtaposed in certain fashion. For example: the above given tree T can be thought of as made up of "three" 2-stars (or 2-paths) such that second 2-star is attached to pendant point of first 2-star while third 2-star is attached to pendant point of second 2-star. This should help us to construct lattice path for the tree representing the graceful avatar of the tree under consideration.

**Remark 6.12:** Pushing a lattice path in n-delta lattice downwards to fit in (n+1)-delta lattice is called **shifting**. Shifting of the lattice path in n-delta lattice downwards to fit it in (n+1)-delta lattice doesn't change the isomorphism type of the tree. The only effect of this shifting is that the new tree in the bigger lattice contains separations {2, 3, ….., n} and so not graceful and has some missing vertex among the vertices {1, 2, ….., n}.

**Example:** When the lattice path
(1, 7)→(1, 6)→(1, 5)→(2, 5)→(3, 5)→(3, 4) in 7-delta lattice is shifted downwards to fit in 8-delta lattice it becomes
(1, 8)→(1, 7)→(1, 6)→(2, 6)→(3, 6)→(3, 5)
Note that separation "1" and vertex with label "4" are absent in this shifted path but this new path represents the tree isomorphic to graceful tree represented by old path.

We now begin to develop a **recursive proof** for the graceful tree conjecture.

**Theorem 6.1:** Every (n+1, n)-tree can be gracefully labeled by suitably modifying the lattice path of its pendant point deleted (n, n-1)-sub-tree in the n-delta lattice (representing graceful avatar of this (n, n-1)-sub-tree) by shifting this lattice path in downward direction in (n+1, n)-delta lattice and **doing surgery at appropriate place to add one lattice point** appropriately such that the modified lattice path will represent the graceful avatar of given (n+1, n)-tree.

**Proof:** We prove the theorem by induction on n the number of vertices. We assume by induction that every tree containing n vertices has an associated lattice path in n-delta lattice representing its graceful avatar. We show that the same is true for tree with (n+1) vertices and we can build a lattice path in (n+1)-delta lattice by shifting the lattice path corresponding to its (n, n-1)-sub-tree and carrying out appropriate surgery on it.

**Step1:** We obtain case n = 3 from case n = 2. For n = 2 the tree is a single edge represented by 2–delta lattice containing single lattice point and the associated lattice

(1, 2)

For n = 3, the lattice is

(1, 2)  (2, 3)
(1, 3)

Up to isomorphism there is only 2-star exists in 3-delta lattice. Here, we shift from the lattice path consisting of one lattice point (1, 2) to (1, 3) and add lattice point (1, 2) or (2, 3) and thus achieve the lattice path (1, 3)→(1, 2), or (1, 3)→(2,3). Thus, (2, 1)-sub-tree represented by lattice path (1, 2) now changes to lattice path (1, 3)→(1, 2), or (1, 3)→(2, 3). Similarly, we obtain case n = 4 from case n = 3. For n = 3 the tree is a 2-star represented by lattice path containing two lattice points in the associated 3–delta lattice

(1, 2)  (2, 3)
(1, 3)

represented either by lattice paths (1, 3)→(1, 2), or (1, 3)→(2, 3).

For n = 4 the 4-delta lattice is

(1, 2)   (2, 3)   (3, 4)
(1, 3)   (2, 4)
(1, 4)

As stated we shift the lattice paths, either (1, 3)→(1, 2) or (1, 3)→(2, 3), whichever considered, to (1, 4)→(1, 3) or (1, 4)→(2, 4), and obtain desired (4, 3) tree as (1, 4)→(1, 3)→(1, 2) or (1, 4)→(1, 3)→(2, 3).

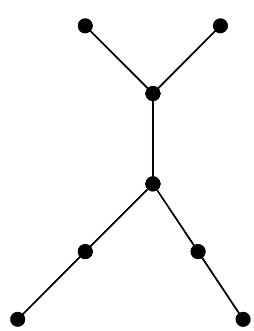
**Step 2:** Consider some (n+1, n)-tree. Delete some of its pendant vertex which will result in an (n, n-1)-sub-tree of given tree. By induction we have a lattice path in n-delta lattice representing its graceful avatar. Shift this lattice path downwards to fit in (n+1)-delta lattice. Draw and label this tree and locate the point where we need to reattach an edge to get the tree isomorphic to given tree. Note that since the (n+1, n)-tree is to be regained from its pendant vertex deleted (n, n-1)-sub-tree by properly reattaching the deleted edge at proper vertex connecting the deleted pendant vertex the lattice path corresponding to this sub-tree must remain present in disguise in the modified lattice path representing desired (n+1, n)-tree. Now, if deleted edge should be attached to vertex with label "1" of (n, n-1)-tree then the part of shifted lattice path going diagonally upwards joining lattice points (1, n+1)→(1, n)→…. →(1, k) needs extension to have part of lattice path as (1, n+1)→(1, n)→…. →(1, k)→(1, k-1) and the other lattice path should continue with same shape to reach first row at some lattice point of (n+1)-delta lattice. So, in this way determine the exact vertex label where one needs modification leading to addition of a lattice point resulting in addition of appropriate edge at appropriate location and modify the lattice path at that lattice point representing appropriate pair of vertices to include the new pair. This surgery should result in the lattice path in (n+1)-delta lattice representing the given tree in graceful avatar.

□

**Corollary:** Graceful tree conjecture is true.

□

**Example:**



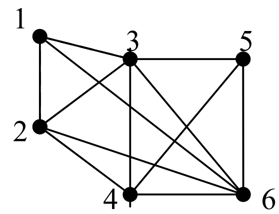
T T*

Let T be the tree for which we are seeking a lattice path in 8-delta lattice. We delete a pendant point and obtain tree T*, a sub-tree of T. We find (By induction) lattice path for T* in 7-delta lattice, namely,
(1, 7)→(1, 6)→(1, 5)→(2, 5)→(3, 5)→(3, 4).
We shift in downwards to fit it in 8-delta lattice, so that we have
(1, 8)→(1, 7)→(1, 6)→(2, 6)→(3, 6)→(3, 5)
We carry out appropriate surgery on this path to include missing edge in T to get lattice path for T in 8-delta lattice as
(1, 8)→(1, 7)→(1, 6)→(2, 6)→(2, 5)→(4, 6)→(3, 4).
Thus we replaced part of path (3, 6)→(3, 5) by (2, 5)→(4, 6)→(3, 4).
Note that the lattice path for tree T* is actually present in lattice path for T as (1, 8)→(1, 7)→(1, 6)→(2, 6)→(4, 6)→(3, 4) as it has to be and gets modified by insertion of a lattice point representing edge (2, 5).

**7. An Algorithm for Arbitrarily Degree Constrained Graceful Tree:**
In this section we proceed with an algorithm for finding arbitrarily degree constrained graceful spanning tree in a given connected labeled graph G containing p vertices and q lines. By Remark 6.1, a p-delta lattice is essentially a representation for complete graph on p vertices where these vertices are labeled by numbers {1, 2, 3, …, p}. A connected (p, q) graph, G, labeled with vertex labels {1, 2, 3, …, p}, is essentially a subgraph of complete graph on p vertices, labeled with vertex labels {1, 2, 3, …, p}, with certain edges which are absent in G. With this we can associate a modified delta lattice, defined below, with any connected labeled graph by just deleting the lattice points and their associated pairs representing edges that are absent in the given connected labeled graph G under consideration.

**Definition 7.1:** A **modified delta lattice (modified n-delta lattice)** is a triangular shaped lattice of points, derived from usual delta lattice (n-delta lattice) associated with a complete graph on p vertices, by deleting the lattice points and their associated vertex pairs representing those edges that are absent in the given connected labeled (p, q) graph G labeled with vertex labels in the set {1, 2, 3, …, p}.

Consider following connected labeled graph G on 6 vertices:




G

The associated “modified 6-delta lattice” for G will be as follows:

(1,2)  (2,3)  (3,4)  (4,5)  (5,6)

(1,3)  (2,4)  (3,5)  (4,6)

(3,6)

(2,6)

(1,6)

Modified 6-delta lattice associated with G

**Algorithm 7.1:**

**Step 1:** Find the corresponding modified p-delta lattice associated with the given connected labeled graph, say, G.

**Step 2:** If this modified p-delta lattice contains a row without containing any lattice point then declare that the graph under consideration doesn’t contain any gracefully labeled tree and stop. Else, go to step 3.

**Step 3:** Let k be the highest degree that defines the constraint for the arbitrarily degree constrained graceful spanning tree to be searched in G. Then try to form a lattice path by starting with lattice point associated with vertex pair (1, p) and go on extending the lattice path in the upward direction by incorporating one lattice point from each row, such that at each stage of selection

(a) The number of distinct vertex labels incorporated in totality belonging the selected vertex pairs m (associated with the successively chosen lattice points) is greater than or equal to (m+1)

(b) The number of lattice points, chosen to incorporate at each stage of selection, on any diagonal or off-diagonal line of the modified p-delta lattice is less than or equal to k.
(c) Continue this way till one arrives at first row.

**Remark 7.1:** If we can continue our selection till we reach the first row satisfying the conditions (a), (b), (c) mentioned above and form the desired (p,p-1) connected graph then we have formed the desired arbitrarily degree constrained graceful spanning tree in the given labeled graph.

**Remark 7.2:** It is important to note that if we drop the condition of graceful labeling and form a connected (p,p-1) graph by choosing pairs in the p–delta lattice as the edges of a graph by imposing only one condition, namely condition (b) given above, then we get arbitrarily degree constrained spanning tree of the given labeled graph, which may not be graceful.

**Remark 7.3:** If the degree of arbitrarily degree constrained graceful spanning tree is constrained to be less than or equal to two then the given (p, q) graph G should be such that its associated modified p-delta lattice should contain at least one zigzag lattice path for its existence in G.

**Problem 7.1:** Find complexity of Algorithm 7.1.

**Problem 7.2:** As explained in Remark 6.4, an (n,n-1)-tree is either a star, a path, or a particular structured arrangement of stars and paths of varied sizes joined together in a particular way at some predefined vertices. Obtain something like a structure theorem for trees which will define every tree in a unique way through this structured representation (one may call it star-path representation, or SP representation of trees.) Use this representation to find the associated lattice path to get its graceful representation in the associated n-delta lattice.

**Problem 7.3:** For given connected labeled (p, q) graph find all the graceful spanning trees that exists in it.

**Problem 7.4:** For given a connected unlabeled (p, q) graph discover the labeling for its vertices using labels {1,2,…,p} which will contain maximum number of gracefully labeled spanning trees.

**Acknowledgements**

Thanks are due to Dr. M. R. Modak, Dr. T.T. Raghunathan, and Mr. Riko Winterle, for their keen interest.

**References**

1. Ringel G., problem 25 (in) Theory of Graphs and its Applications, (Proc. Symp. Smolenice 1963, Czech. Acad. Sci., 162, 1964.
2. West Douglas B., Introduction to Graph Theory, Pages 69-73, Prentice-Hall of India, Private Limited, New Delhi 110-001, 1999.
3. Rosa A..,. On certain valuations of the vertices of a graph. (In) Theory of Graphs, Intl. Symp. Rome 1966, Gordon and Breach, Dunod, 349-355, 1967.
4. Narsingh Deo, Graph Theory with Applications to Engineering and Computer Science, Page 141, Prentice-Hall of India, Pvt. Ltd., New Delhi-110001, 1989.
5. Marthe Bonamy, A Small Report on Graph and Tree Isomorphism, 2010. http://perso.ens-lyon.fr/eric.thierry/Graphes2010/marthe-bonamy.pdf