Survey on Plane 3-Tree with Nearest Neighbor Interchanges and Chordal Bipartite Graph

Tamanna Afroze
M. Sc. Student
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

ABSTRACT

Graph theory is an ancient branch of engineering. Many problems of real-world have been solved by graph theory's principles. In this survey paper, I want to present the plane 3-tree's concept with an interesting branch of another topic of graph theory, that is, chordal bipartite graph. Throughout the development of this survey paper, we will present definitions of chordal graph, bipartite graph, tree, plane 3-tree and different operations of tree architecture. Then I will show a new way to look at the tree architecture with nearest neighbor interchanges. Nearest Neighbor Interchanges is a mechanism that deals with the operation of relative nodes in a tree architecture. Relative nodes are those nodes which come in a same branch or in different branch of a tree which do not interrupt the way from leaf to root with other nodes in the same branch or in another branch are used in the nearest neighbor interchanges to interchange or exchange.We have many applications in which we can apply this nearest neighbor interchange mechanism. The main application is in DNA matching, DNA synthesizing and ribosome particles analysis. I will clearly describe these features in detail in the specified sections of this survey paper. Next I want to show that plane 3-tree has passed the planarity criteria, that is, plane 3-tree is a planar graph and it has straight-line drawing. So, we can construct a bipartite, chordal and chordal bipartite graph which is relevant with the given plane 3-tree. The novelty of this survey paper comprised of several definitions, graphical illustrations of different graph operations and chordal bipartite equivalent of a plane 3-tree.

Keywords

Graph, Tree, Planarity, Nearest Neighbor Interchanges, Chordality, Bipartite graph, Plane 3-Tree.

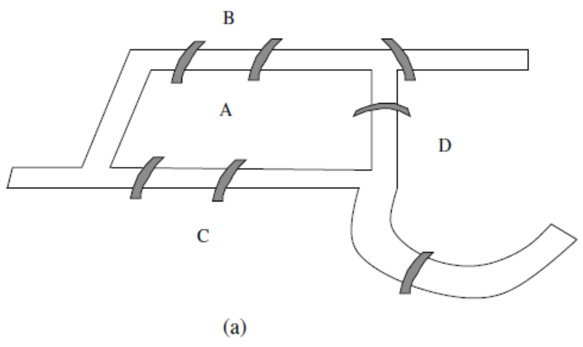
1. INTRODUCTION

A graph consists of a set of vertices and set of edges, each joining two vertices. Usually an object can be represented by a vertex and a relationship between two objects is represented by an edge. Thus a graph may be used to represent any information that can be modeled as objects and relationships between those objects. Graph theory deals with study of graphs. The foundation stone of graph theory was laid by Euler in 1736 by solving a puzzle called Konigsberg seven-bridge problem. Konigsberg is an old city in Eastern Prussia lies on the Pregel river. The Pregel river surrounds an island called Kneiphof and separates into two brances as shown in Fig. 1(a) where four land areas are created: the island A, two river banks B and C , and the land D between two branches. Seven bridges connect the four land areas of the city. It is said that the people of Konigsberg used to entertain themselves by trying to devise a route around the city which would cross each of the seven bridges just once. Since their attempts had always failed, many of them beleived that the task was impossible, but there was no proof until 1736. In that year, one of the leading mathematician of that time, Leonhard Euler published a solution to the problem that no such walk is possible. He not only dealt with this particular problem, but also gave a general method for other problems of the same type. Euler constructed a mathematical model for the problem in which each of the four lands A, B, C and D is represented by four points and each of the seven bridges is represented by a curve or a line segment as illustrated in Fig. 1.1(b). The problem can now be stated as follows: Beginning at one of the points A, B, C and D, is it possible to trace the figure without traversing the same edge twice? The matematical model constructed for the problem is known as a graph model of the problem. The points A, B, C and D are called vertices, the line segments are called edges, and the whole diagram is called a graph.

A graph G is a tuple (V,E) which consists of a finite set V of vertices and a finite set of edges; each edge is an unordered pair of vertices. The two vertices associated with an edge e is called the end-vertices of e. We often denote by (u, v), an edge between two vertices u and v. We also denote the set of vertices of a graph G by V (G) and the set of edges of G by E(G). Let e = (u, v) be an edge of a graph G. Then the two vertices u and v are said to be adjacent in G and the edge e is said to be incident to the vertices u and v. The vertex u is also called a neighbor of v in G and vice versa.

Designing graphs have many practicle applications. In this paper I am going to describe some of the necessity of graph theory's concept in solving many real world problems. One of this is phylogenetic trees. Simultaneously plane 3-tree is another planar graph applied for solving planarity related problems.

The reconstruction of evolutionary trees from data sets on overlapping sets of species is a central problem in phylogenetics. Provided that the tree reconstructed for each subset of species is rooted and that these trees together consistently, the space of all parent trees that 'display' these trees was recently shown to satisfy the



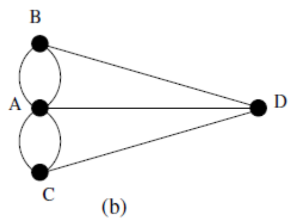


Fig. 1. Graph Model for Konigsberg.

following strong property: there exists a path from any one parent tree to any other parent tree by a sequence of local rearrangements (nearest neighbour interchanges) so that each intermediate tree also lies in this same tree space. However, the proof of this result uses a non-constructive argument. In this paper we describe a special, polynomial-time procedure for navigating from any given parent tree to another while remaining in this tree space. The results are of particular relevance to the recent study of 'phylogenetic terraces'.

Testing the planarity of a graph and possibly drawing it without intersections is one of the most fascinating and intriguing algorithmic problems of the graph drawing and graph theory areas. Although the problem per se can be easily stated, and a complete characterization of planar graphs has been known since 1930, the first linear-time solution to this problem was found only in the 1970s.

Planar graphs play an important role both in the graph theory and in the graph drawing areas. In fact, planar graphs have several interesting properties: for example, they are sparse and 4-colorable, they allow a number of operations to be performed more efficiently than for general graphs, and their inner structure can be described more succinctly and elegantly. From the information visualization perspective, instead, as edge crossings turn out to be the main reason for reducing readability, planar drawings of graphs are considered clear and comprehensible.

A central goal in systematic biology is to reconstruct and analyze a (phyloge- netic) tree to describe the evolutionary relationships among present-day species, based on a comparison of their genetic data. This activity has accelerated greatly in recent years due to the rapid advances in new genomic sequencing technology. While biologists in the 1970s might have reconstructed a tree for a dozen species using a single gene, today, phylogenetic trees are routinely con- structed for hundreds or thousands of species, often based on large numbers (hundreds or thousands) of genes. These trees reveal how species today trace back to a common ancestor by displaying the branching pattern and timing of separation events. These trees, in turn, provide insights into how particular evo- lutionary innovations arose that are present in the group of species under study (e.g. multicellularity, photosynthesis, wings, large brains, etc). Phylogenetic trees can also shed light on the amount of biodiversity captured by different subsets of species and how much of this biodiversity may be at risk from extinc- tion in the near future (a recent example is the analysis in of the estimated tree for all $\tilde{1}0,000$ species of birds).

Tree reconstruction methods often attempt to combine the evolutionary signal across many different genes. One of the problems with such an approach is that each gene may be present in only a subset of the species. This may be because the gene simply does not exist in some species or because the gene, though present, is yet to be sequenced for those species. Moreover, the set of species that lack a given gene typically varies from gene to gene. Patchy taxon coverage has a direct combinatorial consequence for tree reconstruction methods, which often seek to optimize (e.g. minimize) some objective function based on how well the data each tree. The result can be large collections of equally-optimal trees (i.e. a at landscape of trees), that form a (phylogenetic) 'terrace'.

A straight-line grid drawing of a plane graph G is a planar drawing of G, where each vertex is drawn at a grid point of an integer grid and each edge is drawn as a straight-line segment. The height, width and area of such a drawing are respectively the height, width and area of the smallest axis-aligned rectangle on the grid which encloses the drawing. A minimum-area drawing of a plane graph G is a straight-line grid drawing of G where the area is the minimum. It is NP-complete to determine whether a plane graph G has a straight-line grid drawing with a given area or not.

Separator are things to separate a connected graph. Separator can be an edge, a vertex or a subgraph. Many separator results for topological graphs, especially for planar embedded graphs base on the fact that separators have a structure that cuts the surface into two or more pieces onto which the separated subgraphs are embedded on. The celebrated and widely applied (e.g., in many divide-and-conquer approaches) result of Lipton and Tarjan finds in planar graphs a small sized separator. However, their result says nothing about the structure of the separator, it can be any set of discrete points. Applying the idea of Miller for finding small simple cyclic separators in planar triangulations, one can find small separators whose vertices can be connected by a closed curve in the plane intersecting the graph only in vertices, so-called Jordan curves. Tree-decompositions have been historically the choice when solving NP-hard optimization and FPT problems with a dynamic programming approach. Although much is known about the combinatorial structure of tree-decompositions, no result is known to the author relating to the topology of tree-decompositions of planar graphs. A branch-decomposition is another tool, that was introduced by Robertson and Seymour in their proof of the Graph Minors Theorem and the parameters of these similar structures, the treewidth tw(G) and branchwidth bw(G) of the graph G.Recently, branch-decompositions started to become a more popular tool than tree-decompositions, in particular for problems whose input is a topologically embedded graph.

A graph having no cycles is said to be acyclic. A forest is an acyclic graph. A tree is a connected graph without any cycles, or a tree is a connected acyclic graph. The edges of a tree are called branches. It follows immediately from the definition that a tree has to be a simple graph (because self-loops and parallel edges both form cycles). Figure 2 displays all trees with fewer than six vertices.

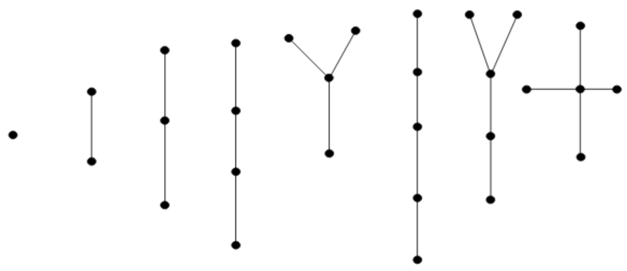


Fig. 2. Illustrating a tree structure.

A graph is a tree if and only if there is exactly one path between every pair of its vertices.The various kinds of data structures referred to as trees in computer science have underlying graphs that are trees in graph theory, although such data structures are generally rooted trees. A rooted tree may be directed, called a directed rooted tree, either making all its edges point away from the rootin which case it is called an arborescence, branching, or out-tree, or making all its edges point towards the rootin which case it is called an anti-arborescence or in-tree. A rooted tree itself has been defined by some authors as a directed graph.

2. PRELIMINARIES

In this section I am going to elaborately discuss the graphs, its several properties, different ytpes of graphs and application of graphs.

2.1 Graph

A graph G is a tuple consisting of a finite set V of vertices and a finite set E of edges where each edge is an unordered pair of vertices. The two vertices associated with an edge e is called the end-vertices of e. We often denote by (u, v), an edge between two vertices u and v. We also denote the set of vertices of a graph G by V (G) and the set of edges of G by E(G). We generally draw a graph G by representing each vertex of G by a point or a small circle and each edge of G by a line segment or a curve between its two end-vertices. For example, Fig. 2.1 represents a graph G where V (G) = {v1, v2, . . . , v11} and E(G) = {e1, e2, . . . , e17}. We often denote the number of vertices of a graph G by n and the number of edges of G by m; that is, n = V (G) and m = E(G). We will use these two notations n and m to denote the number of vertices and the number of edges of a graph unless any confusion arises. Thus n = 11 and m = 17 for the graph in Figure 3.

A loop is an edge whose end-vertices are the same. Multiple edges are edges with the same pair of end-vertices. If a graph G does not have any loops or multiple edges, then G is called a simple graph; otherwise it is called a multigraph. The graph in Figure 3 is a simple graph since it has no loops or multiple edges. On the other hand, the graph in Figure 4 contains a loop e5 and two sets of multiple edges {e2, e3, e4} and {e6, e7}. Hence the graph is a

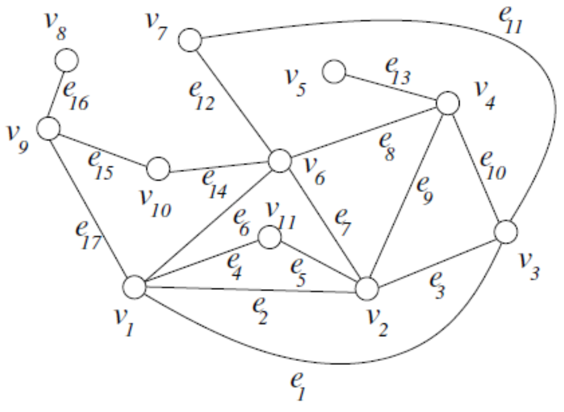


Fig. 3. Graph with 11 vertices and 17 edges.

multigraph. In the remainder of the book, when we say a graph, we shall mean a simple graph unless there is any possibility of confusion.

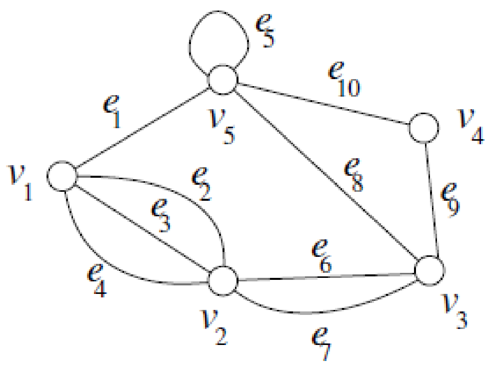


Fig. 4. Multigraph.

We call a graph a directed graph or a digraph if edge is associated with a direction, as illustrated in Figure 5(a). One can consider a directed edge as a one-way street. We thus can think an undirected graph as a graph where each edge is directed in both direction. We call a graph an weighted graph if an weight is assigned to each vertex or each edge. Figure 5(b) illustrates an edge-weighted graph.

A subgraph of a graph G = (V,E) is a graph G = (V ,E) such that V V and E E. If G contains all the edges of G that join vertices in V , then G is called the subgraph induced by V .

2.1.1 Adjacencies, Incidence and Degree.

Let e = (u, v) be an edge of a graph G. Then the two vertices u and v are said to be adjacent in G and the edge e is said to be incident to the vertices u and v. The vertex u is also called a neighbor of v in G and vice versa. In the graph in Figure 3, the vertices v1 and v3 are adjacent; the edge e1 is incident to the vertices v1 and v3. The neighbors of the vertex v1 in G are v2, v3 v6, v9 and v11.

The degree of a vertex v in a graph G, denoted by deg(v) or d(v), is the number of edges incident to v in G, with each loop at v counted

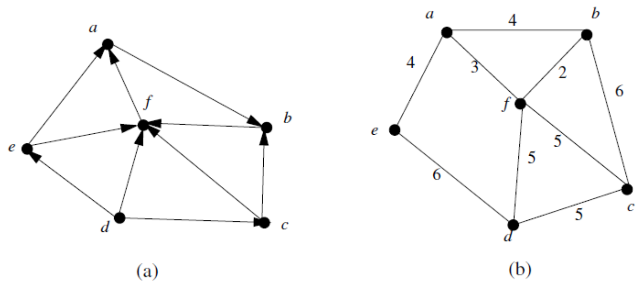


Fig. 5. (a) A directed graph and (b) an edge-weighted graph.

twice. The degree of the vertex v1 in the graph of Figure 4 is 5. Similarly, the degree of the vertex v5 in the graph of Figure 4 is also 5.

Since the degree of a vertex counts its incident edges, it is obvious that the summation of the degrees of all the vertices in a graph is related to the total number of edges in the graph.

2.2 Different Graph Definition

In this subsection we will discuss different types of graphs which will come in discussion in the subsequent sections of the paper.

2.2.1 Null Graph. A graph with an empty edge set is called a null graph. A null graph with n vertices is denoted by N_n. Figure 6(a) illustrates the null graph N_6 with six vertices. A null graph is a subgraph of any graph with the same number of vertices.

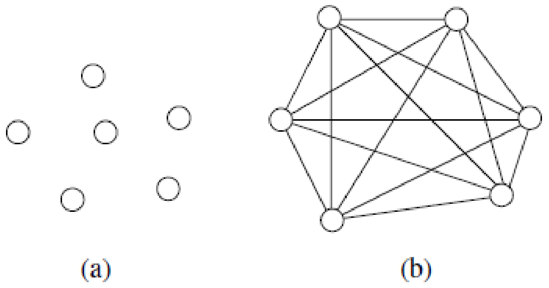


Fig. 6. (a) A null graph N_6 with six vertices, (b) a complete graph K_6 with six vertices.

2.2.2 Complete Graph. A graph in which each pair of distinct vertices are adjacent is called a complete graph. A complete graph with n vertices is denoted by K_n. It is trivial to see that Kn contains n(n 1)/2 edges. Figure 6(b) illustrates a complete graph K_6 with six vertices. Any graph is a subgraph of the complete graph with the same number of vertices and thus the number of edges in a graph with n vertices is at most n(n 1)/2.

2.2.3 Independent set and Bipartite Graph. Let G = (V,E) be a graph. A subset of vertices V V is called an independent set in G if for every pair of vertices u, v V , there is no edge in G joining the two vertices u and v. A graph G is called a bipartite graph if the vertex set V of G can be partitioned into two disjoint non-empty sets V1 and V2, both of which are independent. The two sets V1 and V2 are often called the partite sets of G. Each edge of a bipartite graph G thus joins exactly one vertex of V1 to exactly one vertex of V2. Figure 7 shows two bipartite graphs where the independent partitions are highlighted in both the graphs. Given a graph G, one can test whether G is a bipartite graph in a naive approach by considering each possible bipartition of the vertices of G and checking whether the two partitions are independent or not. However since there are 2^n 2 possible bipartition of a graph with n vertices, this approach

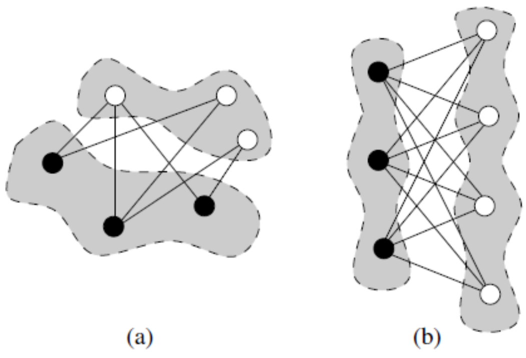


Fig. 7. Two bipartite graphs: the two independent partite sets are highlighted for each of them, one containing the black colored vertices and the other containing the white colored vertices.

takes exponential time. Fortunately, there is a linear-time algorithm to test whether a graph is bipartite or not. The idea is simple. Using a breadth first search (BFS) on the graph G, color the vertices of G with two colors such that no two adjacent vertices receive the same color. We say colors of two vertices conflict if the vertices are adjacent and receive the same color. If a conflict-free coloring can be done by BFS, then G is bipartite, otherwise not.

Let G be a bipartite graph with the two independent sets V1 and V2. We call G a complete bipartite graph if for each vertex u V1 and each vertex v V2, there is an edge (u, v) in G. Figure 7(b) illustrates a complete bipartite graphs where the two partite sets contains 3 and 4 vertices, respectively. This graph is denoted by $K_{3,4}$. In general, a complete bipartite graph is denoted by $K_{m,n}$ if its two partite sets contains m and n vertices, respectively. One can easily see that $K_{m,n}$ contains mxn edges.

2.2.4 Chordal Graph. A chord in a cycle is an edge which goes between two vertices which are not consecutive in the cycle. A graph G is chordal if there are no chordless cycles in G of length greater than three. Figure 8 illustrates a chordal graph of nine vertices. Chordal graphs always contain a vertex v such that the neighborhood of v is a clique; such a vertex is called a simplicial vertex.

A chordal graph is an undirected graph with the property that every cycle of length greater than three has a chord (an edge between nonconsecutive vertices in the cycle). Chordal graphs have attracted interest in graph theory because several combinatorial optimization problems that are very difficult in general turn out to be easy for chordal graphs and solvable by simple greedy algorithms. Examples are the graph coloring problem and the problem of finding the largest clique in a graph. Chordal graphs have been studied extensively since the 1950s and their

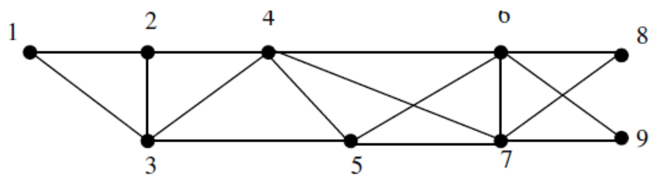


Fig. 8. Chordal Graph.

history shares some key events with the history of semidefinite optimization. In particular, it was Shannons 1956 paper that led Berge to the definition of perfect graphs, of which chordal graphs are an important subclass, and Lovsz to one of the most famous early applications of semidefinite optimization.

Chordal graphs in applications often result from graph elimination, a process that converts a general undirected graph into a chordal graph by adding edges. Graph elimination visits the vertices of the graph in a certain order, called the elimination order. When vertex v is visited, edges are added between the vertices that are adjacent to v, follow v in the elimination order, and are not yet mutually adjacent. If no edges are added during graph elimination the elimination order is called a perfect elimination order. It has been known since the 1960s that chordal graphs are exactly the graphs for which a perfect elimination order exists. A variety of algorithms based on different forms of variable elimination can be described and analyzed via graph elimination. Examples include the solution of sparse linear equations (Gauss elimination), dynamic programming (eliminating optimization variables by optimizing over them), and marginalization of probability distributions (eliminating variables by summation or integration). Variable elimination is a natural approach in many applications and this partly explains the diversity of the disciplines in which chordal graphs have been studied.

2.2.5 Tree. A tree is an undirected graph G that satisfies any of the following equivalent conditions:

—G is connected and has no cycles.

—G is acyclic, and a simple cycle is formed if any edge is added to G.

—G is connected, but is not connected if any single edge is removed from G.

—G is connected and the 3-vertex complete graph K3 is not a minor of G.

—Any two vertices in G can be connected by a unique simple path.

If G has finitely many vertices, say n of them, then the above statements are also equivalent to any of the following conditions:

—G is connected and has n 1 edges.

—G has no simple cycles and has n 1 edges.

As elsewhere in graph theory, the order-zero graph (graph with no vertices) is generally excluded from consideration: while it is vacuously connected as a graph (any two vertices can be connected by a path), it is not 0-connected (or even (1)-connected) in algebraic topology, unlike non-empty trees, and violates the ”one more vertex than edges” relation.

An internal vertex (or inner vertex or branch vertex) is a vertex of degree at least 2. Similarly, an external vertex (or outer vertex, terminal vertex or leaf) is a vertex of degree 1.

An irreducible tree (or series-reduced tree) is a tree in which there is no vertex of degree 2.

A rooted tree T is a tree in which one of the vertices is distinguished from the others. The distinguished vertex is called the root of the tree T and every edge of T is directed away from the root. If v is a vertex in T other than the root, the parent of v is the vertex u such that there is a directed edge from u to v. When u is the parent of v, v is called a child of u. A vertex in T , which has no children, is called a leaf. Any vertex which is not a leaf in T is an internal vertex. A descendant of u is a vertex v other than u such that there is a directed path from u to v. Let i be any vertex of T . Then we define a subtree $T(i)$ rooted at i as a subgraph of T induced by vertex i and all the descendants of i. An ordered rooted tree is a rooted tree where the children of any vertex are ordered counter-clockwise.

2.2.6 Planar Graph. A drawing is planar if no two distinct edges intersect except, possibly, at common endpoints. A graph is planar if it admits a planar drawing. A planar drawing partitions the plane into connected regions called faces. The unbounded face is usually called external face or outer face. If all the vertices are incident to the outer face, the planar drawing is called outerplanar and the graph admitting it is an outerplanar graph. Given a planar drawing, the (clockwise) circular order of the edges incident to each vertex is fixed. Two planar drawings are equivalent if they determine the same circular orderings of the edges incident to each vertex (sometimes called rotation scheme). A (planar) embedding is an equivalence class of planar drawings and is described by the clockwise circular order of the edges incident to each vertex. A graph together with one of its planar embedding is sometimes referred to as a plane graph.

2.2.7 Connected Graph. An undirected graph G is connected if, for each pair of nodes u and v, G contains a path from u to v. A graph G with at least k + 1 vertices is k-connected if removing any k 1 vertices leaves G connected. Equivalently, by Mengers theorem, a graph is k-connected if there are k independent paths between each pair of vertices. 3-connected, 2- connected, and 1-connected graphs are also called triconnected, biconnected, and simply connected graphs, respectively. It is usual in the planarity literature to relax the definition of biconnected graph so to include bridges, i.e., graphs composed by a single edge between two vertices. A separating k-set is a set of k vertices whose removal disconnects the graph. Separating 1- and 2-sets are called cutvertices and separation pairs, respectively. Hence, a connected graph is biconnected if it has no cutvertices and it is triconnected if it has no separation pairs.

2.2.8 Traingulated Plane Graph. If all the faces of a plane graph G are triangles, then G is called a triangulated plane graph. For a cycle C in a plane graph G, we denote by G(C) the plane subgraph of G inside C (including C).A maximal planar graph is one to which no edge can be added without losing planarity. Thus in any embedding of a maximal planar graph G with n 3, the boundary of every face of G is a triangle, and hence the embedding is often called a triangulated plane graph. Although a general graph may have up to n(n 1)/2 edges, it is not true for planar graphs. If G is a planar graph with n(3) vertices and m edges, then m 3n 6. Moreover the equality holds if G is maximal planar.

2.3 Different Graph Operation

In this subsection we will give an illustration of different graph operations with figures showing the steps of a particular operation.

A (rooted) tree T is a connected acyclic graph with one distinguished vertex, called the root r. A spanning tree of a graph G is a tree T such that V (T) = V (G) and E(T) E(G).
Given two graphs G1(V1,E1) and G2(V2,E2), their union G1 ł G2 is the graph G(V1 ł V2,E1 ł E2). Analogously, their intersection G1 G2 is the graph G(V1 V2,E1 E2). A graph G2 is a subgraph of G1 if G1 ł G2 = G1.
Given a graph G(V,E) and a subset V of V , the subgraph induced by V is the graph G(V ,E), where E is the set of edges of E that have both endvertices in V . Given a graph G(V,E) and a subset E of E, the subgraph induced by E is the graph G(V ,E), where V is the set of vertices incident to E. A subdivision of an edge (u, v) consists of the insertion of a new node w and the replacement of (u, v) with edges (u,w) and (w, v). A graph G2 is a subdivision of G1 if it can be obtained from G1 through a sequence of edge subdivisions.

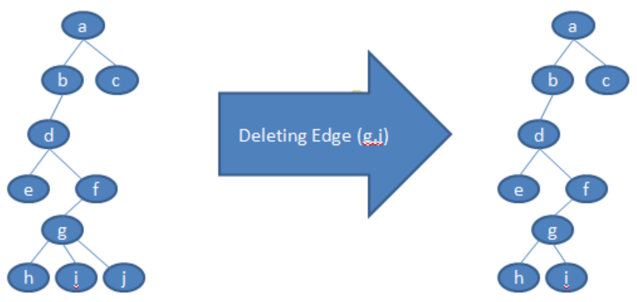


Fig. 9. Deleting Edges From Graph.

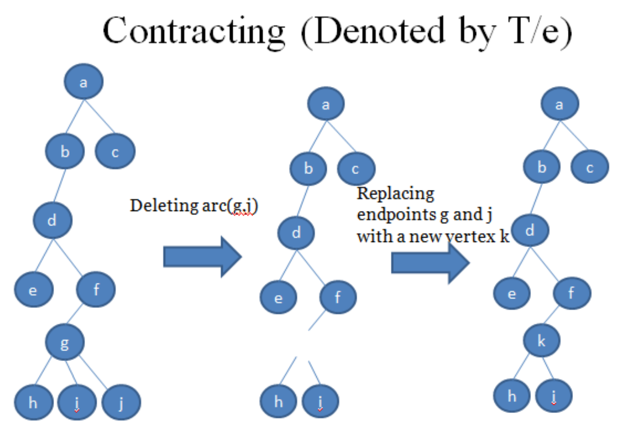
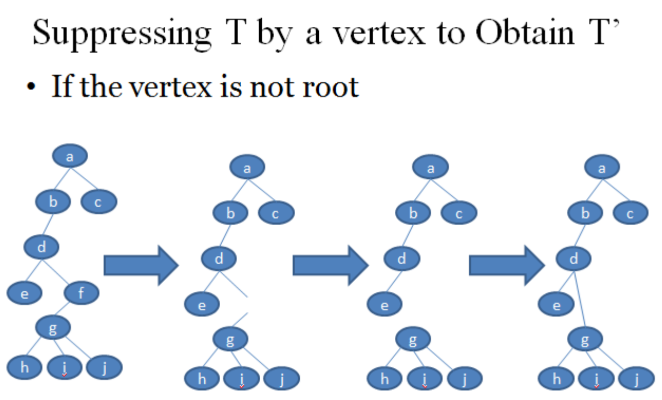
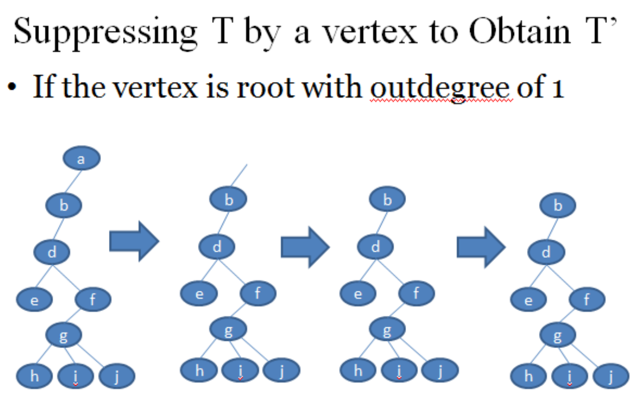

Contracting (Denoted by T/e)

Fig. 10. Contracting by a Vertex in a Graph.


Suppressing T by a vertex to Obtain T'

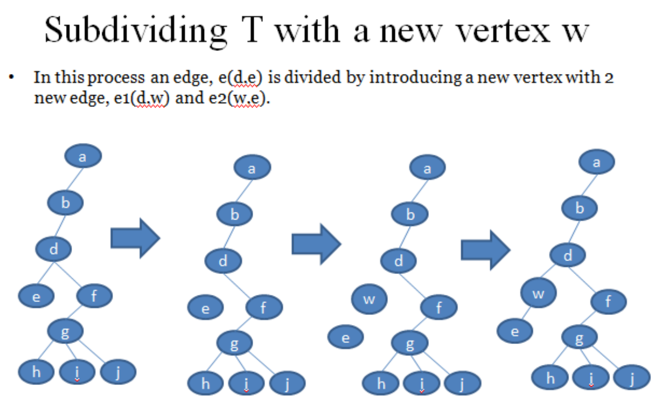
- If the vertex is not root

Fig. 11. Suppressing by a Vertex in a Graph when vertex is not a root.


Suppressing T by a vertex to Obtain T'

- If the vertex is root with outdegree of 1

Fig. 12. Suppressing by a Vertex in a Graph when vertex is a root.


Subdividing T with a new vertex w

- In this process an edge, e(d,e) is divided by introducing a new vertex with 2 new edge, e1(d,w) and e2(w,e).

Fig. 13. Subdividing by a Vertex in a Graph.

3. ROOTED SUBTREE PRUNED AND REGRAFT (RSPR)

According to [1], let $T \in RB(X)$ and let e = (v, u) be an arc of T. We say that $T^1 \in RB(X)$ is an rSPR-neighbour of T if T^1 can be obtained from T by the following procedure. Let t_u be the subtree of T rooted at u. Delete arc e, pruning the subtree t_u. To regraft t_u, either: (i) Choose an arc f of the tree generated by deleting t_u from T and subdivide f with a vertex w, then insert the arc (w, u), regrafting the subtree t_u, or (ii) Introduce a vertex$r$, insert the arc (r,r_T) where r_T is the root of T, and then insert the arc (r , u), regrafting the subtree t_u . Note that r is the root of the resulting tree. Lastly, suppress v. We have now obtained a tree T^1 that is an rSPR-neighbour of T.

3.1 rSPR Illustrated by Figures

Pruning a graph by deleting an edge. This is the first operation of pruning simply showing a pruned tree.

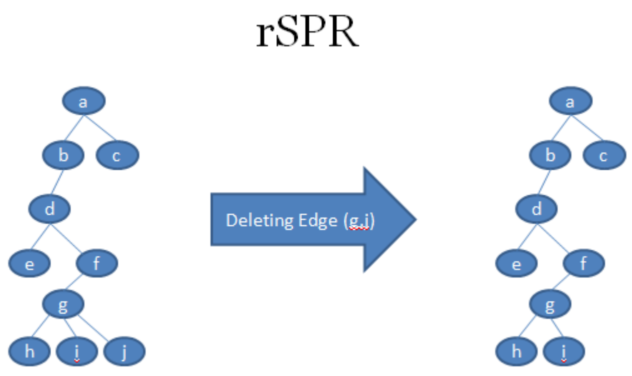


Fig. 14. Pruning by Edge Deleting in a Graph.

Pruning a graph by deleting a subtree. This is an extended method of pruning to get a pruned tree.

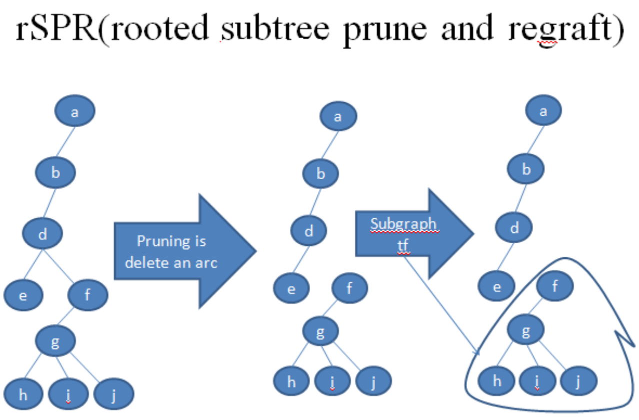


Fig. 15. Pruning by Subtree Deletion in a Graph.

Regrafting procedure to obtain a regraft graph from the given graph. This is the first method of regrafting.

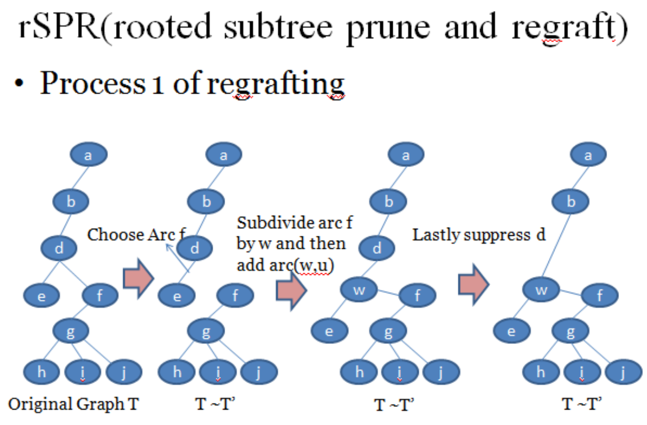

rSPR(rooted subtree prune and regraft)
• Process 1 of regrafting
Choose Arc f
Subdivide arc f by w and then add arc(w,u)
Lastly suppress d
Original Graph T
T ~T'
T ~T'
T ~T'

Fig. 16. Process 1 of Regrafting in a Graph.

Regrafting procedure to obtain a regraft graph from the given graph. This is the first method of regrafting.

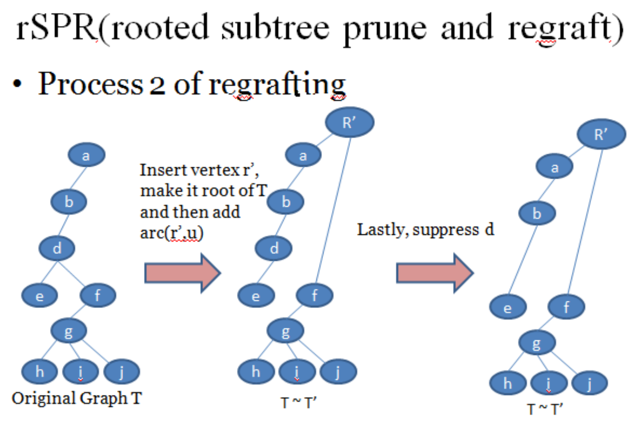

rSPR(rooted subtree prune and regraft)
• Process 2 of regrafting
Insert vertex r', make it root of T and then add arc(r',u)
Lastly, suppress d
R'
Original Graph T
T ~ T'
T ~ T'

Fig. 17. Process 2 of Regrafting in a Graph.

4. NEAREST NEIGHBOR INTERCHANGES

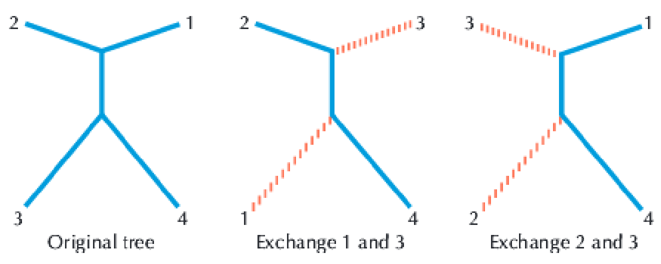
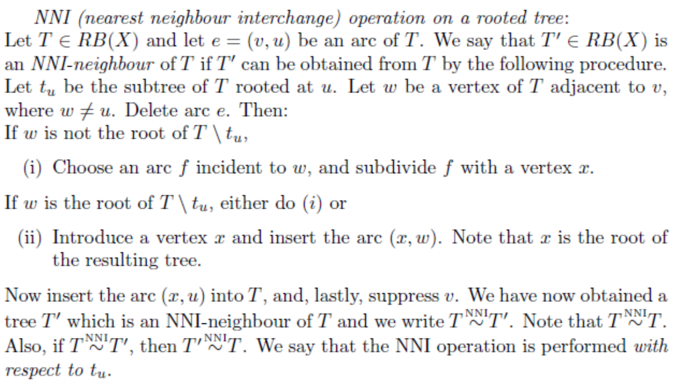


Fig. 18. Nearest Neighbor Interchanges.

Nearest neighbor interchange is a method of generat ing alternat ive trees in which an internal branch in a tree is selected and then the subtrees that are connected to that bra nch are exchanged.

When used for tree searching, such as in parsimony methods , each tree would be assigned a score and the tree with the better score wou ld serve as the start ing point for further analysis. According to [1] nearest neighbor interchanges are defined as following way:



NNI (nearest neighbour interchange) operation on a rooted tree:
Let $T \in RB(X)$ and let $e = (v, u)$ be an arc of T. We say that $T' \in RB(X)$ is an *NNI-neighbour* of T if T' can be obtained from T by the following procedure. Let t_u be the subtree of T rooted at u. Let w be a vertex of T adjacent to v, where $w \neq u$. Delete arc e. Then:
If w is not the root of $T \setminus t_u$,

(i) Choose an arc f incident to w, and subdivide f with a vertex x.

If w is the root of $T \setminus t_u$, either do (i) or

(ii) Introduce a vertex x and insert the arc (x, w). Note that x is the root of the resulting tree.

Now insert the arc (x, u) into T, and, lastly, suppress v. We have now obtained a tree T' which is an NNI-neighbour of T and we write $T \overset{\text{NNI}}{\sim} T'$. Note that $T \overset{\text{NNI}}{\sim} T$. Also, if $T \overset{\text{NNI}}{\sim} T'$, then $T' \overset{\text{NNI}}{\sim} T$. We say that the NNI operation is performed *with respect to* t_u.

Fig. 19. NNI

5. PLANE 3-TREE

A plane graph G with $n \geq 3$ vertices is called a plane 3-tree mentioned in [2] if the following (a) and (b) hold: (a) G is a triangulated plane graph; (b) if $n > 3$, then G has a vertex x whose deletion gives a plane 3-tree G of n 1 vertices.

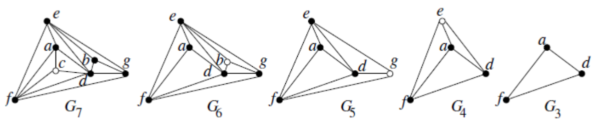


Fig. 20. Examples of Plane 3-tree

Note that, vertex x may be an inner vertex or an outer vertex of G. We denote a plane 3-tree of n vertices by G_n. Examples of plane 3-trees are shown in Figure 20; G_6 is obtained from G_7 by removing the inner vertex c of degree three. Then G_5 is obtained from G_6 by deleting the inner vertex b of degree three. G_4 is obtained from G_5 by deleting the outer vertex g of degree three and G_3 is obtained in a similar way.
Let G_n is a plane 3-tree with vertices $n > 3$. Then G_n satisfies following two conditions. (a) G_n has one inner vertex 'x' of degree 3 removal of which produces a plane 3-tree witn n-1 vertices, G_{n-1}. (b) G_n has *exactly* one inner vertex 'y' which is neighbor of 3 outer vertices of G_n.
Any plane 3-tree G_n, $n > 3$, there is exactly one inner vertex 'y' which is the common neighbor of all the outer vertices of G_n. We call vertex 'y' the representative vertex of G_n.
Let G_n be a plane 3-tree with $n > 3$ vertices and C be any triangle of G_n. Then the subgraph G_n(C) is a plane 3-tree.
Let p be the representative vertex and a, b, c be the outer vertices of G_n. The vertex p, along with the three outer vertices a, b and c, form three triangles {a, b, p}, {b, c, p} and {c, a, p} as illustrated in Figure 21. We call those three triangles the nested triangles around p.

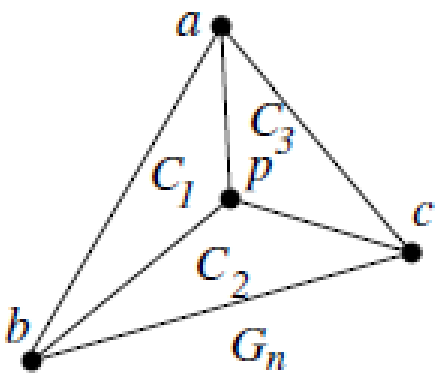


Fig. 21. Nested triangles around p.

The representative tree of G_n is an ordered rooted tree T_{n-3} satisfying the following two conditions (a) and (b). (a) if n = 3, T_{n-3} consists of a single vertex. (b) if $n > 3$, then the root p of T_{n-3} is the representative vertex of G_n and the subtrees rooted at the three counter-clockwise ordered children q1, q2 and q3 of p in T_{n-3} are the representative trees of G_n(C1), G_n(C2) and G_n(C3), respectively, where C1, C2 and C3 are the three nested triangles around p in counter-clockwise order.

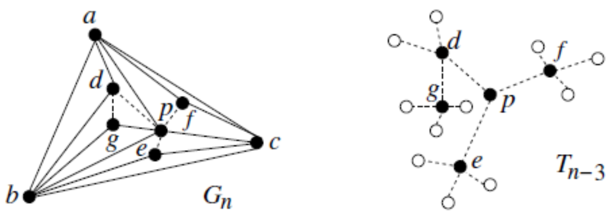


Fig. 22. representative tree T_{n-3} of G_n.

Figure 22 shows the representative tree T_{n-3} of G_n. Every plane 3-tree has a unique representative tree with exactly n-3 internal vertices and 2n-5 leaves. From Figure 22 we can see that representative tree T_{n-3} of G_n has 5 internal vertices and 11 leaves when original graph n=8 vertices. For any plane 3-tree representative tree T_{n-3} of G_n can be found in O(n) time.

6. MY PROPOSITION

6.1 Chordal Bipartite Graph

Chordal bipartite graphs are a useful and natural class of graphs with many interesting properties. We will see that these properties give rise to a form of representation which is fairly natural and is space optimal, but is not implicit. Chordal bipartite graphs are also interesting in that they are one of the most important classes of graphs which are known to have more than $2^{\Omega(nlogn)}$ and at most $2^{o(n^2)}$members on n vertices.
A graph G is chordal bipartite if G is bipartite, and every cycle of length at least 6 has a chord. The name comes from the idea that these are the bipartite analogue of chordal graph. G is chordal if every cycle which can have a chord (that is, every cycle of length

at least 4) has a chord. In bipartite graphs, only cycles of length at least 6 can have chords, so chordal bipartite graphs are bipartite graphs such that every cycle which could have a chord actually has a chord.
Chordal bipartite graphs are analogus to chordal graphs in many other ways.It is known that G is chordal if and only if evry minimal seperator is a clique, and that a bipartite graph is chordal bipartite if and only if every minimal separator is a complete bipartite graph.

6.2 Plane 3-tree with Nearest Neighbor Interchange and Chordal Bipartite Graph Synthesis

In this section I want to show how a chordal bipartite graph can be drwan as a plane 3-tree and its nearest neighbor interchanges affect the representative tree.

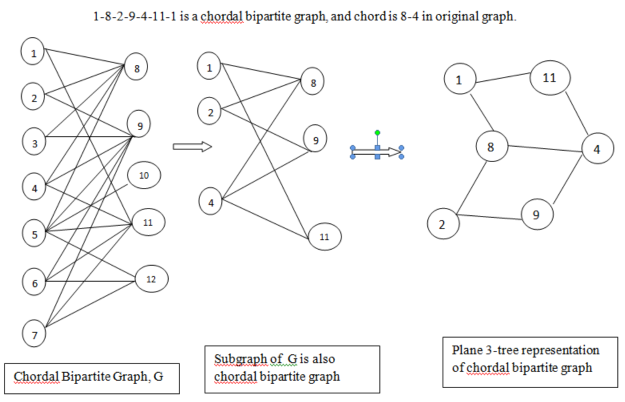

1-8-2-9-4-11-1 is a chordal bipartite graph, and chord is 8-4 in original graph.
Chordal Bipartite Graph, G
Subgraph of G is also chordal bipartite graph
Plane 3-tree representation of chordal bipartite graph

Fig. 23. Illustration with figure.

With nearest neighbor interchanges we can develop a edge adding mechanism calculating appropriate scoring function.
I have measured the scoring function based on the links with the neighbor connected with each other node. In this way we can have the maximum score for the center nodes. We can measure maximum likelihood of the nodes for particle synthesis in DNA matching problem. From figure 23 we can identify the maximum score for node 4. Node 4 has maximum score of 4. So, maximum likelihood function can be applicable for node 4. In this way, we can draw figures for DNA synthesis and can add vales to the colored images for each particle. These colored factor can be further diagnosed in image analysis part which I have considered separately in another design.
Plane 3-tree with chordal bipartite graph has many more interesting properties, but I have mentioned only a portion of work in this section. Because writing of this work focuses on investigating how plane 3-tree can be affected by nearest neighbor interchanges and chordal bipartite graph which is useful in DNA sample matching and ribosome particle synthesis.

7. CONCLUSION

In this paper, I wanted to show how 3 types of graph can be joined with a tree property to give a better understanding of some graph properties which can be used to analyse bioinformatics research. Calculating scoring function with nearest neighbor interchanges we can improve our DNA matching problem, ribosome synthesis operation. The scoring function can be designed with specific information suitable for a particular problem. This is a survey on different types of graphs and mechanisms of different types of operation on graphs. While doing work on this survey I have identified this new technique for chordal bipartite graph and plane 3-tree. These combined mechanism is useful for graph partitioning in DNA and ribosome synthesis of bioinformatics research.

8. REFERENCES

[1] Sarah Mark, Jeanette C. McLeod and Mike Steel. *A navigation system for tree space*. Journal of Graph Algorithms and Applications (JGAA),vol.20, pp247-268,April 2016.
[2] Debajyoti Mondal Rahnuma Islam Nishat Md. Saidur Rahman Muhammad Jawaherul Alam. *Minimum-Area Drawings of Plane 3-Trees* Journal of Graph Algorithms and Applications (JGAA),2009.
[3] Md Saidur Rahman. *Basic Graph Theory* 2002.
[4] Md Saidur Rahman. *Planar Graph Drawing* 2004.
[5] J. P. Spinrad. *Efficient Graph Representations, American Mathematical Society* 2003.