Journal of Physical Sciences, Vol. 10, 2006, 188 – 200

An Encounter with Graphs

Rajat Kumar Pal and Samar Sen Sarma
Department of Computer Science and Engineering
University of Calcutta
92, A. P. C. Road, Kolkata - 700 009

Received 13 October, 2006 ; accepted 15 November, 2006

ABSTRACT

So far diagnosis of some problems we came across during our works with algorithms, graphs played significant roles. In this paper, we have tried to show that graphs really occupy a major role in computer science and engineering. The abstraction of problems as different graph models as well as graphs in digital system design methodology allows us to have a close look or refresher outlook on graphs. We have also touched root level, historical purview, graphs in contemporary Indian scenario, and a significant aspect of perfect graphs in VLSI area. We observe that hard computing does not suit in most of graph problems. Soft computing approaches are often a tool in graph algorithm design. A pessimal algorithm also came across our route that is really an intellectual property to share with the readers. To part, with we remind that no diagnosis is final and that pathology is in fact always eager to have a new look.

1. Prologue [1, 2, 5, 6, 11, 13, 21, 23, 24]

This paper is an honest, documented cursory glance to application and interest in graph theory for last about 300 years. This is merely an introduction and invitation. We hope the taste will linger after presentation of the text and certain audience may even take the help of the references for further studies. We will be rewarded if we have inspired somebody a little. We announce the prologue and retire for the time being, after us better players will arrive.

2. In Lieu of an Introduction [1, 2, 5, 6, 11, 13, 21, 23, 24]

A backtrack to the History of Graphs: Graph theoretic folklores are so widely publicized that natural human inclination will insist on seeing evolution of graph algorithms in reference to these folklores. Hence we have an excuse (not lame in any way) for the backtracking.

The four fathers of graph theory Euler, Kirchhoff, Cayley and Hamilton (in chronological order) discovered graph theory while trying to solve either puzzle or a problem of the physical world. The graph theory was christened as a mathematical discipline in 1736 with the first paper of Euler on this subject. He solved (or rather proved it unsolvable) the *Konigsberg Bridge Problem* by translating it in terms of graphs. Next 100 years were almost a blank period for graph theory. In 1847, Kirchhoff developed the theory of trees in order to solve the electrical network equations. Although a physicist, he ably abstracted electrical networks like mathematician in terms of graphs.

Ten years later the number of isomers of saturated hydrocarbons (C_nH_{2n+2}) was expressed by Arthur Cayley as the graphical problem of enumerating the class of *trees* in which each point is of degree 1 or 4. An elegant and general solution was first obtained by Polya, who developed powerful enumeration technique. Extremely effective use of Polya's theorem has been made by the theoretical physicists R. W. Ford and G. E. Uhlenbech in (1956) solving several enumeration problems for graphs which arises in a study of statistical mechanics.

A game invented by Sir William Hamilton in 1859 uses a regular dodecahedron whose 20 vertices are labeled with the name of famous cities. The player is challenged to travel around the world by finding a close circuit along the edges that passes through each vertex exactly once. In fact, in terms of graphs it demanded the necessary and sufficient condition for the existence of Hamiltonian circuits in an arbitrary graph. The most famous unsolved problem in graph theory is the celebrated Four Color Conjecture. It states that four colors are sufficient to color any atlas (a map on a plane) such that no countries with common boundaries have same colors. The conjecture has an interesting history. In a letter to Sir W. R. Hamilton dated 23rd October 1852, Professor Augustus de Morgan mentioned the problem and also printed out the name of his student Cutwise as its originator. The conjecture appeared in print for the first time in June 1878, in the form of a question posed by the well-known mathematician Cayley. Cayley again stated the problem in the first volume of proceedings of the Royal Geographical Society in 1879. The most recent work on this problem was done by Hakim and Apple of Illinois, USA, in 1976.

In 1920, Graph theory arose to a new height due to Konig and others. The real formalization of graph theory was due to Veblen's work.

In 1932, Faster established the connection between geometric networks and electric circuits. It was followed by Whitney's fundamental work on graph theory.

The pace of research in graph theory has accelerated even further in the past two decades.

One of the manifestations of this activity was the immediate success of the Journal of Combinatorial Theory founded in 1966. Perhaps the most striking evidence of this upsurge of interest was the first book ever written in graph theory by Konig (1936). No books were published within 1968. Several books and thousands of papers have been published in graph theory in recent years.

3. Graph Models

We define graph $G = (V,E)$ as a set of vertices and a set of edges. The sets are related by Handshaking theorem stating sum of degrees of vertices a graph is twice the number of edges.

a) Structure of a system

We describe the structure of a system as the abstract graph consisting of its block diagram. For example the block diagram of an Exclusive-OR logic circuit can be represented as a graph as shown below.

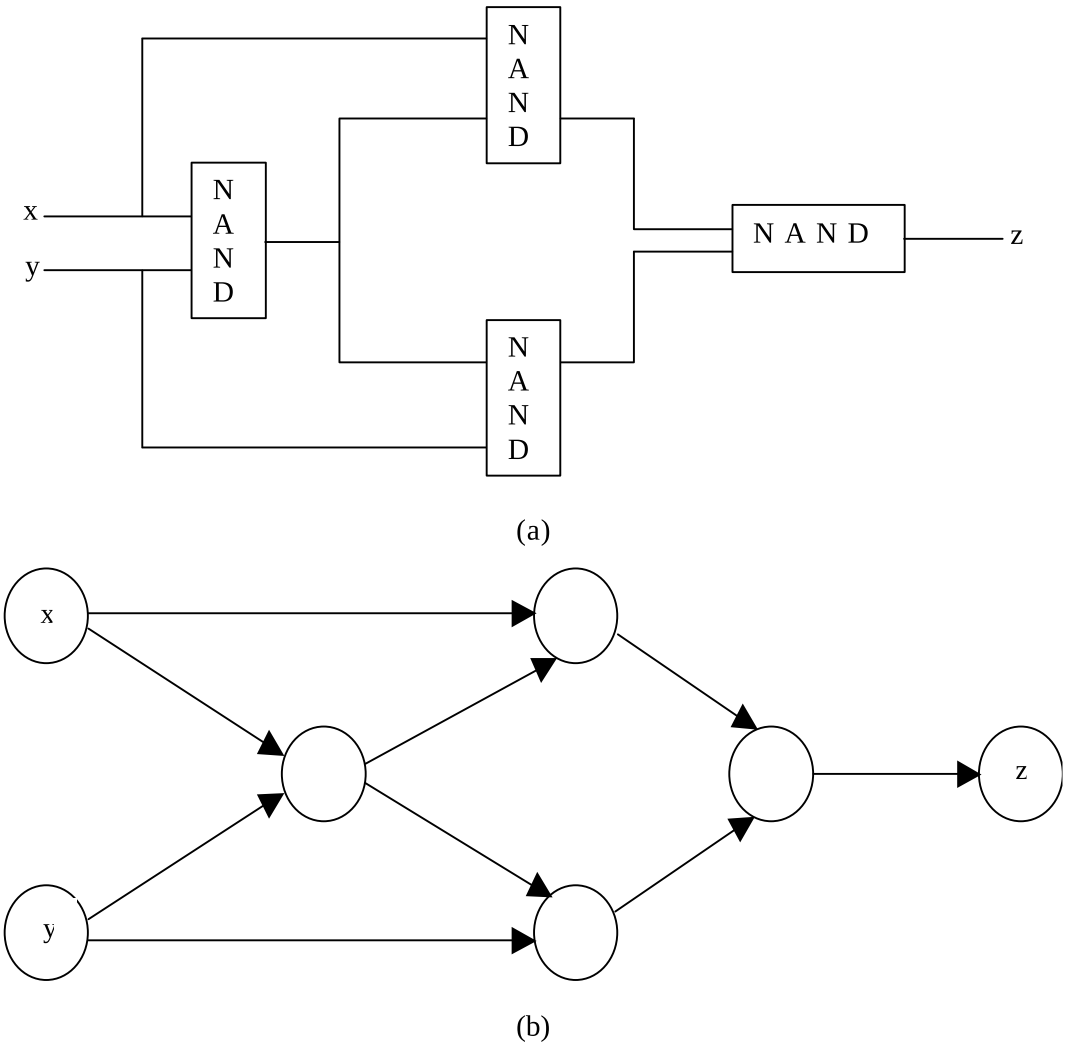

Figure 1: (a) The system and **(b)** the corresponding graph.

The drawing is one to one and all digital system structure can be modeled as a graph.

b) Acquaintance Graph [21]

Represent some people by vertices of a graph and edges to show whether they know each other .The Graph can show how many people in a group are mutual friends or strangers.

c) The Hollywood Graph [21]

Here vertices represent Actors and edges to represent whether they have acted together in a film. According to the Internet Movie database, in November 2001 the Hollywood graph had 5,74,724 vertices representing actors who have appeared in 2,92,609 films, and had more than 16 million edges.

d) Round-Robin Tournaments [21]

A tournament where each team plays each other team exactly once is called a Round-Robin Tournament. Directed graphs where each team is represented by a vertex can model the situation. (a,b) is an edge if team (a) beats team (b).

e) Call Graph [21]

Graphs can be used to model telephone calls made in a network, such as a long distance telephone network. In particular, a directed multigraph can be used to model calls where a vertex represents each telephone number and each telephone call is represented by a directed edge.

f) Graphs and concurrent processing [21]

Computer programs can be executing certain statements concurrently. It is important not to execute depended statements in parallel. The dependence of the statements on previous statements can be represented by a directed graph.

Many other applications like computer network, chemistry, computer programming, signal flow graphs exist where graphs play a role model for solution.

4. Graph Theory in India

The senior author started work on graph theory in the University of Calcutta under Prof. Arun K. Choudhury in Computer Center as early as 1974. However active research started in the sixties through R. C. Bose, Prof. S. S. Shrikhande disproving one conjecture of Euler. Around 1970, ISI Calcutta started work in Graph Theory. IIT Madras (Dr. K. R. Parthasarathy and others), Bombay University (Prof. V. N. Bhatnayak and others), and Sourashtra (Dr. E. Sampathkumar and others). Around 1976, Karnataka University, Dharward (Dr. E. Sampathkumar and his students), Karnataka University P. G. Centre, Gulbarka (Dr.

V. R. Kulli and his students), National College, Tiruchirapalli, Tamil Nadu (Dr. R. Balakrisdhnan and his students), and Mehta Research Institute, at present, Manonmaniam Sundaranar University, Around 1980, IIT Kharagpur (Prof. G.R. Bhattacharjee, R. K. Ghosh, P. Gupta, M. Pal) has started worked on graph algorithm.

Tiruneveli (Dr. S. Arumugam and his students), Cochin University of Science and Technology, Cochin (Dr. A. Vijayakumar and his students), Vidyasagar University (Dr. M. Pal) are actively engaged in graph theoretic research.

Apart from these Research Centers, several persons are doing graph theoretic research.

In 1973, a conference on graph theory was held at Delhi. During December 1976, a symposium on graph theory was held at ISI, Calcutta conducted a 'Seminar on Combinatorics and Applications.'

The following are some Graph Theory seminars conducted more recently.

a) R. C. Bose Memorial Conference on Combinatorial Mathematics and applications, I.S.I., Calcutta, Dec. 14-16, 1988 (Convener: Prof. A. R. Rao).

b) National Seminar on Graph Theory, Annamalai University, Feb. 21-28, 1990 (Convener: Prof. R. Balkrishnan).

c) National Seminar on Graph Theory and its Applications, Belgium, Mar. 26-28, 1991 (Convener: Dr. H. B. Waliker).

d) Symposium on Graph Theory and Combinatorics, Cochin University, May 17-19, 1991 (sponsored by Center for Mathematical Sciences, Trivandrum).

e) Seminar on Graph Theory, Kamaraj University, Madurai, Sep. 11-12, 1992 (Convener: Dr. S. A. Choudam).

f) Seminar on Graph Theory, Annamalai University, Jan. 5-7, 1993 (Convener: Prof. R. Balakrishnan).

g) Seminar on Graph Theory and its Applications, Manonmaniam Sundaranar University, October 17, 1993 (Convener: Prof. S. Arumugam).

h) National Workshop on Graph Theory and its Applications, Manipal Institute of Technology, Manipal, Apr. 4-10, 1994.

i) International Conference on Discrete Mathematics and Number Theory, Tiruchirappalli, Jan. 3-6, 1996 (Convener: Prof. R. Balakrishnan).

j) National Workshop on Graph Theory and its Applications, Manonmaniam Sundaranar University, Thirunelveli, Feb. 21-27, 1996 (Convener: Prof. S. Arumugam).

k) Conference on Graph Connections, Cochin University of Science and Technology, Cochin, Jan. 28-31, 1998 (Convener: Dr. A. Vijayakumar).

l) National Conference on Discrete Mathematics and its Applications, Manonmaniam Sundaranar University, Thirunelveli, Jan. 5-7, 2000 (Convener: Prof. S. Arumugam).

m) Conference on Graph Theory and its Applications, Anna University, Chennai, Mar.14-16, 2001 (Convener: Dr. G. Sethuraman).

n) IWDC Workshops (Now ICDCN: International Conference on Distributed

Computing and Networking) are regularly held in JU, CU, IIM, ISI, IITs, from 1999 onwards.

o) International Workshop on Algorithms, Dec. 15-16, 2005, ISI, Kolkata (Convener: Prof. S. C. Nandy).

In one conference on 2002 (IWDC 2002), N. Deo gave an interesting keynote address in which he used graph theory for finding protein structure and relating it to cancer detection. This is of a new direction in graph theory but still in an embryonic stage.

India certainly has a sizable number of good graph theorists.

Courses on Graph Theory are being at the under-graduate and post-graduate levels in several Indian Universities for students of Mathematics and Computer Science.

5. Graph Algorithms [1, 2, 5, 6, 11, 13, 21, 23, 24

Usually a large varity of problems can be formulated in terms of graphs. For example, scheduling problem as chromatic partitioning, computer networks as graphs of good edge connectivity and vertex connectivity, network analysis in electrical engineering, molecular structure in organic chemistry, program segmentation, verification and testing, geographical information system, any interconnection system, quantum computing, probability theory, VLSI physical design, games and puzzles, in AI and many other problems. The graphs are really large in real life problems. Obviously our aim in graph Algorithm design is optimization of space and time.

a) Connectivity and Distance [6]

The topics include spanning trees, strong and weak connectivity, Transitive closure and shortest path algorithms.

Tree generation algorithms are of two categories -

1. Generation of a single tree,
2. Generation of all trees.

By tree we mean here spanning trees. A single tree of a graph is simple, symmetric and connected. For graph of n vertices tree is a set of $n-1$ edges that touches all vertices. A single tree generation is trivial.

However, Generation of all trees is inherently of exponential time complexity as number of trees in a graph is exponential.

Trees can be generated by divide and conquer and by test and select methods. We have experimented with test and select method and reduce the time complexity appreciably for a large graph.

In case of weighted graph minimal spanning trees can be generated by greedy algorithms, which surprisingly give optimum solution in polynomial time.

Shortest path Algorithm in polynomial time is Dijkstra's Algorithm. This algorithm fails for negative weights. We have some solution with soft computing approach like Genetic Algorithms and simulated annealing.

The cycles in a graph may be of two types -

1. Finding fundamental cycles,
2. Finding all cycles.

From a spanning tree of a graph *f*-cycle can be generated easily. However all cycle generation is exponential in nature. We have worked to generate all cycles with less time complexity than existing algorithms.

b) Some Works on Graph Models for Efficient Routing in Network Environment [20]

We have identified that vertex connectivity and edge connectivity are key issues here. We have chosen standard graphs like Moon Moser graph, De-Bruijn graph, Pascal graph, Critical graph, Hypercube graph for the two key properties. We see that reliability with regular graphs that are sparse in nature is of conflicting constraints. We studied our proposed network with polynomial time complexity of the algorithm. We have initiated a new area of adaptable network topology using regular graphs.

c) Scheduling and Graph Coloring Algorithms

We observe that class scheduling and processor scheduling with optimality criteria can be translated in terms of graphs. Heuristic search and soft computing approach gives a diverse solution with new area.

d) Traveling Salesman Problem (TSP) [10]

We studied this real life problem using soft computing approach. We have seen *TSP* with triangular inequality can be solved with less time for benchmark problem using genetic algorithm approach. In fact, we have used memetic algorithm for such cases. The result is a betterment of solution bound of approximate algorithms. A memetic algorithm with Ant algorithm as basis also generated some interesting results.

e) Graph in VLSI Design [17, 22]

Most of the problems in VLSI physical design are modeled with graphs. The graphs in the area are generally perfect. Perfect graphs have interesting property that the algorithms based on these graphs are polynomial in nature in time complexity.

The initial graphs for channel routing are horizontal and vertical constraint graphs. It is observed that optimal routing solely dependant upon the graphs is NP-hard in nature. The lower band in channel routing can be obtained by using chordal graphs. The other graphs that come to our notice in VLSI physical design are Permutation graphs, Intersection graphs, Circle graphs, Interval graphs, etc. The computation of maximum independent set, minimum coloring, maximum clique detection, etc. are normally done in terms of graphs. Obviously graph algorithms are backbone of the area, as described in brief in the following section.

6. Graph: The Tool to Realize the Problems in VLSI [4, 7-12, 14, 16-19, 22]

From the beginning of its era in the late 1950's, Integrated Circuit (IC) fabrication technology has evolved from being able to integrate a few transistors in *Small Scale Integration (SSI)* to today's integration of several million transistors in *Very Large Scale Integration (VLSI)*. The design of VLSI circuits is a very time consuming and tedious task, though the rapid growth in integration technology in more than last four decades has been made possible by the automation of the various steps involved in design and fabrication of the VLSI chips [22].

Integrated circuits consist of a number of electronic components, built by layering several different materials in a well-defined fashion on a silicon base (or *wafer*). The design of an IC transforms a circuit description into a geometric description, which is known as a *layout*. Usually, a layout consists of a set of planar geometric shapes in several layers, and checked to ensure that it meets all the design requirements defined in system specifications. The process of converting the specification of an electrical circuit into a layout is called the *physical design* [22]. It is an extremely complex and error prone process because of the tight tolerance requirements and the minuteness of the individual components.

Interestingly, most of the problems in VLSI physical design could be modeled into some graph theoretic problems. Often the algorithms that are designed for solving some of these problems are also based on graph theory. In addition, there are special classes of graphs that play a fundamental role in development of these algorithms. Since most of the algorithms in VLSI physical design are based on graph structures, we devote a large portion of this section to graph algorithms for solving some of these problems. Special graphs are also discussed while modeling some of the problems in the domain concerned.

Basic algorithms that are frequently used in physical design can be categorized as two subalgorithms namely graph algorithms and computational geometry based algorithms. Problems related to graphs include graph search, shortest path, minimum spanning tree, and many others. Graph search algorithms include *Depth-First Search (DFS)*, *Breadth-First Search (BFS)*, and *Topological Search*. In DFS, a graph is searched *as deeply as possible*, whereas in BFS, all vertices adjacent to a vertex are explored before exploring any other vertex of the graph. Topological search, based on DFS, computes a topological order, where u appears before v, if (u,v) is a directed edge in the given graph.

Many graph problems are subset selection problems, such as computation of a *Minimum Spanning Tree (MST)* of a graph. Given an edge-weighted graph $G = (V,E)$, select a subset of edges $E' \subseteq E$ such that E' induces a tree and the total cost of edges $\Sigma_{e(i) \in E'} wt(e(i))$, is minimum over all such trees, where $wt(e(i))$ is the weight of the edge $e(i)$. Important algorithms for finding an MST are Kruskal's algorithm [16], Prim's algorithm [19], etc. We are also investigating on algorithms for finding ordered weighted spanning trees in undirected graphs [15].

Many routing problems in VLSI physical design are in essence shortest path problems in special graphs [17]. Shortest path problems, therefore, play a significant role in global and detailed routing algorithms. *Single Pair Shortest Path (SPSP)* problem may be viewed as a vertex or edge selection problem [7]. Precisely, given an edge-weighted graph $G = (V,E)$ and two vertices u, $v \in V$, select a set of vertices $p \subseteq V$ including u, v such that p induces a path of minimum cost in G. *All Pairs Shortest Paths (APSP)* problem is a variant of *SPSP*, in which the shortest path is required for all possible pairs in the graph.

Given an undirected graph $G = (V,E)$, a *matching* is a subset of edges $E' \subseteq E$ such that for all vertices $v \in V$, at most one edge of E' is incident on v. So, a matching computes an independent set of edges in G. A vertex is said to be *matched* by matching E' if some edge in E' is incident on v; otherwise, v is unmatched. A *maximum matching* is a matching with maximum cardinality among all matchings of a graph. A matching is called a *bipartite matching* if the underlying graph is a bipartite graph. Both matching and bipartite matching have many important applications in VLSI physical design. The details of different matching algorithms may be found in [18].

Min-cut and *Max-cut* are two frequently used graph problems that are related to partitioning the vertex set of a graph. The simplest min-cut problem is as follows. Given a graph $G = (V,E)$, partition V into two subsets V_1 and V_2 of equal sizes, such that the number of edges $E' = \{(u,v) \mid u \in V_1, v \in V_2\}$ is minimized. The set E' is also referred to as a *cut*. A generalized min-cut problem may specify the sizes of subsets, or it may require partitioning V into k different subsets. Min-cut and many of its variants have several applications in physical design, including partitioning and placement.

The max-cut problem can be defined as follows. Given a graph $G = (V,E)$, find the maximum bipartite graph of G. Let $G' = (V,E')$ be the maximum bipartite graph of G, which is obtained by deleting k edges from G, then G has a max-cut of size $|E|-k$. It could be noted that both min-cut and max-cut problems are NP-complete in nature [10]. Hadlock [12] presented an algorithm that finds max-cut of a planar graph.

Minimum cost spanning trees and single pair shortest paths are two edge selection problems that can be solved in polynomial time. Surprisingly, a simple variant of these two problems, called the *Steiner minimum tree* problem, is computationally hard. The *Steiner Minimum Tree (SMT)* problem can be defined as follows. Given an edge weighted graph $G = (V,E)$ and a subset $d \subseteq V$, select a subset $V' \subseteq V$, such that $d \subseteq V'$ and V' induces a tree of minimum cost over all such trees. The set d is referred to as the set of *demand points* and the set $V'-d$ is referred to as *Steiner points*. It is easy to see that if $d = V$, then SMT is equivalent to MST; on the other hand, if $|d| = 2$, then SMT is equivalent to SPSP. Unlike MST and SPSP, SMT and many of its variants are NP-complete [9].

Steiner trees arise in VLSI physical design in routing of multi-terminal nets. If two points in a plane are to be connected using a shortest path, the problem is referred to as the problem of routing a two-terminal net. If the net has more than two terminals, then the problem of interconnecting all such terminals using minimum amount of wire length is known as the problem of routing a multi-terminal net that corresponds to the minimization of the total cost of edges in the Steiner tree. The global and detailed routing of multi-terminal nets is an important problem in the layout of VLSI circuits [17]. This problem has traditionally been viewed as a Steiner tree problem [4, 14]. A Steiner tree whose edges are constrained to rectilinear shapes is called a *Rectilinear Steiner Tree (RST)*.

The basic objects in VLSI design are rectangles and lines, and the basic problems in physical design deal with arrangements of these rectangles in a two- or three-dimensional space [22]. The relationship between these objects, such as overlap and distance, are very critical in development of physical design algorithms. Graphs are a well-developed tool used to study relationships between the objects. Naturally, graphs are used to model many VLSI physical design problems and they play a very important role in almost all VLSI design algorithms. Here, in this section, we introduce various graphs that are used in modeling of physical design problems.

Overlap graph, *containment graph*, and *interval graph* are realized when a set of lines are aligned to an axis. If $G_O = (V,E_O)$ is an overlap graph, $G_C = (V,E_C)$ is a containment graph, and $G_I = (V,E_I)$ is an interval graph for the same set of lines aligned to an axis, then it is easy to see that $E_I = E_O \cup E_C$ [20]. Interval graphs form a well-known class of graphs and have been studied extensively [11]. These graphs arise in many routing problems, including single-row routing, channel routing, and over-the-cell routing [22]. If lines are not aligned along an axis, then it is usually assumed (for example, in channel routing) that all the lines originate at a specific y-location and terminate at a specific y-location. This type of diagram is sometimes called a *matching diagram*. *Permutation graphs* are frequently used in routing and can be defined by matching diagram. It is well known that the class of containment graphs is equivalent to the class of permutation graphs [11].

A *channel* is a rectangular routing region consisting of a set of fixed terminals on two of its opposite sides only. Otherwise, the routing region is known as a *switchbox* that contains fixed terminals to be routed on at least three sides of the rectangular region [17]. The channel routing problem that arises rather frequently in VLSI design uses permutation graphs to model the problem. The graph defined by the intersection of lines in a switchbox is equivalent to a *circle graph*. Overlap graphs are equivalent to circle graphs. Circle graphs were originally defined as the intersection graph of chords of a circle can be recognized in polynomial time [8].

As mentioned earlier, rectangles are used to represent circuit blocks in a layout design, where no two rectangles are allowed to overlap. Rectangles may share edges, i.e., two rectangles may be neighbours to each other. Accordingly, a *neighbourhood graph* or *adjacency graph* is obtained, where a vertex represents a rectangular block and an edge between a pair of vertices is introduced if the associated blocks are adjacent. The neighbourhood graph is useful in the global routing phase, where each channel is defined as a rectangle and two channels are neighbours if they share a boundary. Often adjacency is determined when a net connects the two blocks, and a placement is realized with the help of *rectangular dualization* in floorplanning [22].

The classes of graphs used in physical design are related to several well-known classes of graphs, such as *triangulated graphs*, *comparability graphs*, and *co-comparability graphs* [11, 22]. The relations between the graphs are shown in Figure 2. A graph is called *chordal* if every cycle containing at least four vertices has a chord. Chordal graphs are also known as triangulated graphs. A graph is defined as a comparability graph, if it is *transitively orientable*. A graph is called co-comparability graph, if the complement of the graph is transitively orientable.

Triangulated and comparability graphs can be used to characterize *interval graphs*. A graph is called an interval graph if it is triangulated as well as co-comparable. Similarly, comparability and co-comparability graphs can be used to characterize *permutation graphs*. These graphs are comparable as well as co-comparable.

The classes of graphs mentioned above are not unrelated; in fact, interval graphs and permutation graphs have a non-empty intersection. Similarly, the classes of permutation graphs and bipartite graphs have a non-empty intersection. On the other hand, the class of circle graphs properly contains the class of permutation graphs; see Figure 2 for reference.

Several interesting problems related to classes of graphs discussed above arise in VLSI physical design. Some of these problems are computation of *Maximum Independent Set*, *Maximum Clique*, *Minimum Colouring*, etc. It should be noted that most of the problems have polynomial time complexity algorithms for comparability, co-comparability, and triangulated graphs. This is due to the fact that these graphs are *perfect graphs* [11]. Perfect graphs admit polynomial time complexity algorithms for the problems just mentioned above, among many other problems.

Interval graphs and permutation graphs are defined by the intersection of different classes of perfect graphs, and are therefore themselves perfect graphs [11]. As a result, many problems that are NP-hard for general graphs are polynomial time solvable for these graphs. On the other hand, circle graphs are not perfect graphs, and generally speaking are much harder to deal with as compared to interval and permutation graphs.

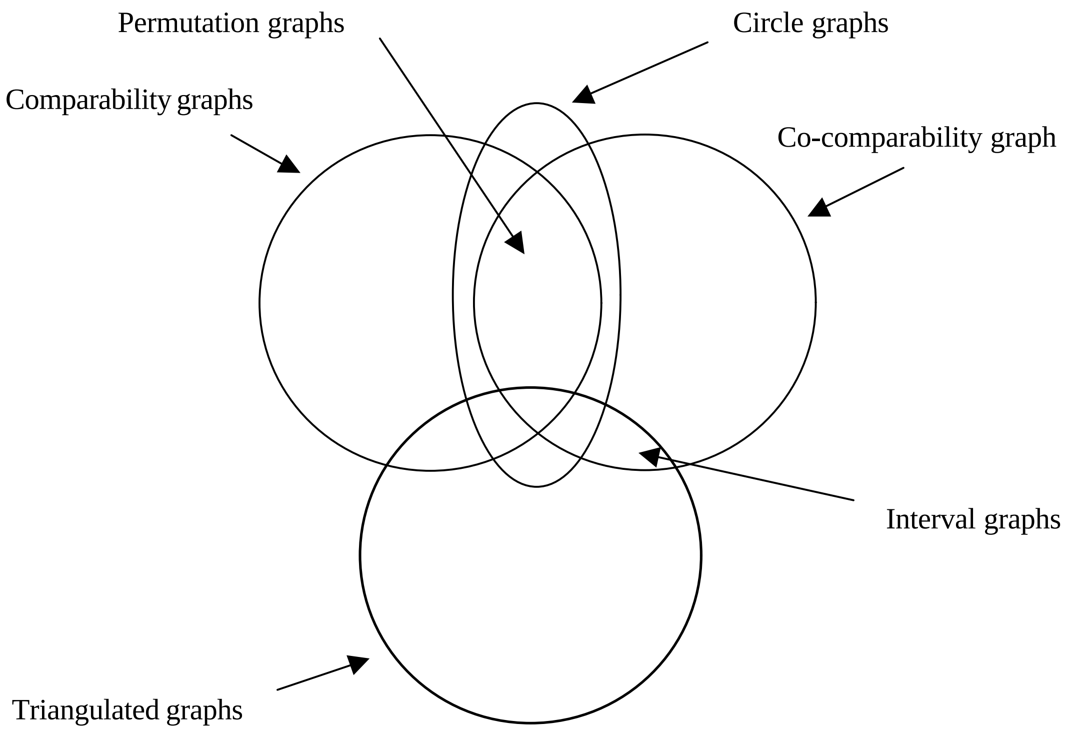

Figure 2: Relationship between different classes of graphs.

7. An Aberration

We are working on pessimal algorithm design for channel routing [3]. It is practical in the sense that we should be slow in sorting something if we are sending to Paris to perform it. Thus it is more affordable to route non-optimally by finding non-trivial lower bound on the number of tracks required in routing a channel. Hard and soft computing approaches using constrained and non-constrained graphs gave us alluring results.

8. Conclusion

Once we encounter an essay *Your Pet Aversions* in our school days. May be it is to write a conclusion of such an article. It is better to be reluctant on such a condition of motion. Obviously, we are at the outskirt of labyrinth. We better depart saying “Carry on baby, Graphs is part of your life.”

REFERENCES

1. Arumaygum S. and S. Ramachandran, Invitation to Graph Theory, SciTech, 2001.
2. Bondy J. A. and U. S. R. Murty, Graph Theory with Applications, American Elsevier, 1976.
3. Broder A. and J. Solfi, Pessimal Algorithms and Simplexity Analysis, ACM SGAGACT NEWS, Vol. 16, No. 3, pp. 49-53, 1984.

4. Chiang C., M. Sarrafzadeh and C. K. Wong, A Powerful Global Router: Based on Steiner Min-Max Trees, *Proc. of IEEE Int. Conf. on Computer-Aided Design*, pp. 2-5, 1989.
5. Clark J. and D. Holton, A First Look at Graph Theory, Allied Publishers, 1995.
6. Deo N., Graph Theory, Prentice-Hall, Inc., 2003.
7. Dijkstra E. W., A Note on Two Problems in Connexion with Graphs, *Numerische Mathematik*, Vol. 1, pp. 269-271, 1959.
8. Gabor C. P., W. L. Hsu and K. J. Supowit, Recognizing Circle Graphs in Polynomial Time, *Proc. of 26th IEEE Symposium on Foundation of Computer Science*, pp. 106-116, 1985.
9. Garey M. R. and D. S. Johnson, The Rectilinear Steiner Tree Problem is NP-Complete, *SIAM Journal of Applied Mathematics*, Vol. 32, pp. 826-834, 1977.
10. Garey M. R. and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.
11. Golumbic M. C., *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, 1980.
12. Hadlock F., Finding a Maximum Cut of a Planar Graph in Polynomial Time, *SIAM Journal of Computing*, Vol. 4, No. 3, pp. 221-225, 1975.
13. Harary F., Graph Theory, Addison-Wesley, 1969.
14. Ho J. M., G. Vijayan and C. K. Wong, A New Approach to the Rectilinear Steiner Tree Problem, *IEEE Trans. on Computer-Aided Design*, Vol. 9, No.2, pp. 185-193, 1985.
15. Kapoor and Ramesh, An Algorithm for Enumerating All Spanning Trees of Undirected and Weighted Graphs, SIAM Journal of Computing, Vol. 24, pp. 247-265, 1997.
16. Kruskal J. B., On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, *Proc. of the American Mathematical Society*, Vol. 7, No.1, pp. 48-50, 1956.
17. Pal R. K., Multi-Layer Channel Routing: Complexity and Algorithms. *NAROSA Publishing House, New Delhi* (Indian Edition, *Paperback*), *CRC Press, Boca Raton, USA* and *Alpha Sc. Intl. Ltd., UK* (International Editions, *Hardbound*), Sep. 2000.
18. Papadimitriou C. H. and K. Steigliz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Inc., 1982.
19. Prim R. C., Shortest Connection Networks and Some Generalizations, *Bell System Technical Journal*, 1957.
20. Reingold, Nievergelt and Deo, Combinatorial Algorithms, Prentice-Hall, Inc., 1977.
21. Rosen K. H., Discrete Mathematics, 2003.
22. Sherwani N. A., *Algorithms for VLSI Physical Design Automation*, Kluwer Academic Publishers, Boston, 1993.
23. West D. B., Introduction to Graph Theory, Prentice-Hall, Inc., 2003.
24. Wilson R. J., Introduction to Graph Theory, Academic Press, 1979.