Chapter 3

Data Structures and Basic Algorithms

VLSI chip design process can be viewed as transformation of data from HDL code in logic design, to schematics in circuit design, to layout data in physical design. In fact, VLSI design is a significant database management problem. The layout information is captured in a symbolic database or a polygon database. In order to fabricate a VLSI chip, it needs to be represented as a collection of several layers of planar geometric elements or polygons. These elements are usually limited to Manhattan features (vertical and horizontal edges) and are not allowed to overlap within the same layer. Each element must be specified with great precision. This precision is necessary since this information has to be communicated to output devices such as plotters, video displays, and pattern-generating machines. Most importantly, the layout information must be specific enough so that it can be sent to the fab for fabrication. Symbolic database captures net and transistor attributes. It allows a designer to rapidly navigate throughout the database and make quick edits while working at a higher level. The symbolic database is converted into a polygon database prior to tapeout. In the polygon database, the higher level relationship between the objects is somewhat lost. This process is analogous to conversion of a higher level programming language (say FORTRAN) code to a lower level programming language (say Assembly) code. While it is easier to work at symbolic level, it cannot be used by the fab directly. In some cases, at late stages of the chip design process, some edits have to be made in the polygon database. The major motivation for use of the symbolic database is technology independence. Since physical dimensions in the symbolic database are only relative, the design can be implemented using any process. However, in practice, complete technology independence has never be reached.

The layouts have historically been drawn by human layout designers to conform to the design rules and to perform the specified functions. A physical design specialist typically converted a small circuit into layout consisting of a set of polygons. These manipulations were time consuming and error prone,
even for small layouts. Rapid advances in fabrication technology in recent years have dramatically increased the size and complexity of VLSI circuits. As a result, a single chip may include several million transistors. These technological advances have made it impossible for layout designers to manipulate the layout databases without sophisticated CAD tools. Several physical design CAD tools have been developed for this purpose, and this field is referred to as Physical Design Automation. Physical design CAD tools require highly specialized algorithms and data structures to effectively manage and manipulate layout information. These tools fall in three categories. The first type of tools help a human designer to manipulate a layout. For example, a layout editor allows designers to add transistors or nets to a layout. The second type of tools are designed to perform some task on the layout automatically. Example of such tools include channel routers and placement tools. It is also possible to invoke a tool of second type from the layout editor. The third type of tools are used for checking and verification. Example of such tools include; DRC (design rule checker) and LVS verifier (layout versus schematics verifier). The bulk of the research of physical design automation has focused on tools of the last two types. However, due to broad range and significant impact the tools of second type have received the most attention. The major accomplishment in that area has been decomposition of the physical design problem into several smaller (and conceptually easier) problems. Unfortunately, even these problems are still computationally very hard. As a result, the major focus has been on development on design and analysis of heuristic algorithms for partitioning, placement, routing and compaction. Many of these algorithms are based on graph theory and computational geometry. As a result, it is important to have a basic understanding of these two fields. In addition, several special classes of graphs are used in physical design. It is important to understand properties and algorithms about these classes of graphs to develop effective algorithms in physical design.

This chapter consists of three parts. First we discuss the basic algorithms and mathematical methods used in VLSI physical design. These algorithms form the basis for many of the algorithms presented later in this book. In the second part of this chapter, we shall study the data structures used in layout editors and the algorithms used to manipulate these data structures. We also discuss the formats used to represent VLSI layouts. In the third part of this chapter, we will focus on special classes of graphs, which play a fundamental role in development of these algorithms. Since most of the algorithms in VLSI physical design are graph theoretic in nature, we devote a large portion of this chapter to graph algorithms. In the following, we will review basic graph theoretic and computation geometry algorithms which play a significant role in many different VLSI design algorithms. Before we discuss the algorithms, we shall review the basic terminology.