Mathematics Monograph Series 26

Combinatorial Theory in Networks

Xu Junming

（组合网络理论）

SCIENCE PRESS
Beijing

Mathematics Monograph Series 26

Xu Junming

Combinatorial Theory in Networks

(组合网络理论)

SCIENCE PRESS
Beijing

Responsible Editor: Li Xin

Published by Science Press
16 Donghuangchenggen North Street
Beijing 100717, P. R. China

Printed in Beijing

ISBN 978-7-03-036478-4

Preface

The advent of very large scale integrated (VLSI) circuit and the development with great speed of the modern communication technology have enabled us to design more complicatedly, more conveniently and economically high performance supercomputers and massive interconnection networks. The modern supercomputers achieve their gains by increasing the number of processing elements, rather than by using faster processors. The most difficult technical problem in constructing a supercomputer will be the design of the interconnection network through which the processors communicate. Selecting an appropriate and adequate topological structure of an interconnection network will become a critical issue, on which considerable research effort has been made over the past decades. This book is aimed to attract the readers' attention to such an important research area.

Graph theory is a fundamental and powerful mathematical tool for designing and analyzing interconnection networks, since the topological structure of an interconnection network is essentially a graph. This fact has been universally accepted by computer scientists and engineers. This book provides the most basic problems, concepts and well-established results on the topological structure and analysis of interconnection networks in the language of graph theory. The material originates from a vast amount of literature, but the theory presented is developed carefully and skillfully. The treatment is generally self-contained, and most stated results are proved.

The book consists of four parts, seventeen chapters. The first part, consisting of two chapters, introduces how to model an interconnection network by a graph and provides a self-contained exposition of the basic graph-theoretic concepts, terminologies, notations and the corresponding backgrounds of networks as well as the basic principles of network design. Some basic results on graph theory used in the book are stated. The second part, consisting of four chapters, presents three major methods for large-scale network design: the line graph method, the Cayley method and the Cartesian product method. The fundamental properties of graphs constructed by these methods are presented in details. As applications of these methods, the third part, consisting of five chapters, provides four classes of the most well-known network structures: the hypercube, the de Bruijn, the Kautz and the double loop networks and their many desirable properties as well. At the last chapter in this part, other common network structures such as mesh, grid, pyramid, cube-connected

cycle, butterfly, omega, and shuffle-exchange networks are simply mentioned. The fourth part, consisting of six chapters, is a main part in the book. It presents some basic issues and research results in analysis of fault-tolerant network consisting of six research aspects, one of each chapter, involving routings, the fault-tolerant diameter, Menger-type problems in parallel systems, the wide diameter, the generalized independence and domination numbers, and restricted fault tolerance, from which the reader can easily find some interesting research issues to study further.

The book is developed from the original version titled *Topological Structure and Analysis of Interconnection Networks*, published by Kluwer Academic Publishers in 2001. During the latest decade, some topics have made a new development and obtained some new results. In this version, we have made a great adjustment on the total framework and enriched some new research results.

Reading the book is not difficult for readers familiar with elementary graph theory and group theory. The book will be useful to those readers who intend to start research in design and analysis of interconnection network structures, and students in computer science and applied mathematics, theoretic computer scientists, engineers, designer of interconnection networks, applied mathematicians and other readers who are interested in network theory.

I am indebted to many friends and colleagues for their interest in and help with this project. A number of people have read the various versions of this book and offered useful comments and advice as a result. In particular, I thank Xu Min, Ma Meijie, Yang Chao, Huang Jia, Zhou Shuming, Hu Futao, He Weihua, Li Xiangjun, Hong Zhenmu. They read through the entire manuscript, corrected many technical errors and provided all graphs. Thank Graduate School of USTC and National Natural Science Funds for providing funds to support my research on these topics involved in this book since 1997.

Finally, I would like to thank my wife, Qiu Jingxia, for her support, understanding and love, without which this work would have been impossible.

Xu Junming
(xujm@ustc.edu.cn)
June 2011
USTC, Hefei

Contents

Part I
Networks and Graphs

A connection pattern of components in a system is called an interconnection network of the system, which can be modeled by a graph. This fact has been universally accepted and used by computer scientists and engineers. Moreover, practically it has been demonstrated that graph theory is a very powerful mathematical tool for designing and analyzing topological structure of interconnection networks.

The first part consists of two chapters. As the topological structure of a network is a graph, in the first chapter, we will briefly recall some basic concepts, notations and main results on graph theory used in this book as well as the corresponding backgrounds in networks. In last section of this chapter, we will use the language of graph theory to introduce some fundamental principles that we should conform to in the process of design of an interconnection network.

According to the basic principles of network design, one desires that designed networks are provided with high regularity and symmetry. In the second chapter, we will introduce such a special class of graphs, called transitive graphs, which possess high regularity and symmetry, thus, is an important and ideal class of topological structures of interconnection networks.

Chapter 1
Fundamentals of Networks and Graphs

In this chapter, we will briefly recall some basic concepts and notations on graph theory used in this book as well as the corresponding backgrounds in networks. Some basic results on graph theory will be stated, but some proofs will be omitted. For a comprehensive treatment of the graph-theoretic concepts and results discussed herein, the reader is referred to any standard text-book on graph theory, for example, Bondy and Murty [59], Chartrand and Lesniak [83], or Xu [503].

1.1 Graphs and Networks

In this section, we will introduce some concepts on graphs as well as how to model an interconnection network by a graph. Although they have been contained in any standard text-book on graph theory, these concepts defined by one author are different from ones by another. In order to avoid quibbling it is necessary to present a formidable number of definitions.

A *graph* G is an ordered pair (V, E), where both V and E are non-empty sets, $V = V(G)$ is the *vertex-set* of G, elements in which are called vertices of G; $E = E(G) \subseteq V \times V$ is the *edge-set* of G, elements in which are called *edges* of G. The number of vertices of G, also called *order* of G, is denoted by $v(G)$. The number of edges of G, also called *size* of G, is denoted by $\varepsilon(G)$.

Two vertices corresponding an edge are called the *end-vertices* of the edge. The edge whose end-vertices are identical is a *loop*. The end-vertices of an edge are said to be *incident* with the edge, and vice versa. Two vertices are said to be *adjacent* if they are two end-vertices of some edge; two edges are said to be *adjacent* if they have an end-vertex in common.

If $E \subseteq V \times V$ is considered as a set of ordered pairs, then the graph $G = (V, E)$ is called a *directed graph*, or *digraph* for short. For an edge e of a digraph G, sometimes, called a *directed edge* or *arc*, if $a = (x, y) \in E(G)$, then vertices x and y are called the *tail* and the *head* of e, respectively; and e is called an *out-going edge* of x and an *in-coming edge* of y.

If $E \subset V \times V$ is considered as a set of unordered pairs, then the graph $G = (V, E)$

is called an *undirected graph*. Note that an undirected graph does not admit loops. Usually, it is convenient to denote an unordered pair of vertices by xy or yx instead of $\{x, y\}$. Edges of an undirected graph are sometimes called *undirected edges*.

A graph G is *empty* if $\varepsilon(G) = 0$, denoted by K_v^c, and *non-empty* otherwise. An undirected graph can be thought of as a particular digraph, a *symmetric digraph*, in which there are two directed edges called *symmetric edges*, one in each direction, corresponding to each undirected edge. Thus, to study structural properties of graphs for digraphs is more general than for undirected graphs. A digraph is said to be *non-symmetric* if it contains no symmetric edges.

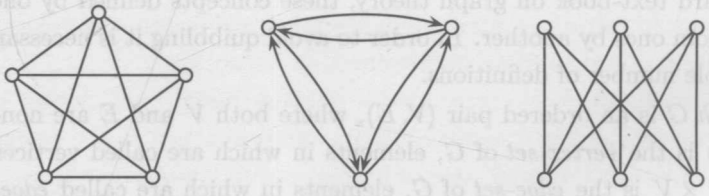
Two graphs G and H are *isomorphic*, denoted by $G \cong H$, if there exists a bijective mapping θ between $V(G)$ and $V(H)$ satisfying the *adjacency-preserving condition*:

$$(x, y) \in E(G) \Leftrightarrow (\theta(x), \theta(y)) \in E(H).$$

The mapping θ is called an *isomorphism* between G and H.

Up to isomorphism, there is just one *complete graph* of order n, denoted by K_n, and one *complete bipartite graph* $G(X \cup Y, E)$, denoted by $K_{m,n}$ if $|X| = m$ and $|Y| = n$, where $\{X, Y\}$ is a bipartition of $V(G)$. It is customary to call $K_{1,n}$ a *star*.

The graphs shown in Figure 1.1 are a complete undirected graph K_5, a complete digraph K_3 and a complete bipartite undirected graph $K_{3,3}$, respectively.



(a) a complete undirected graph K_5 (b) a complete digraph K_3 (c) $K_{3,3}$

Figure 1.1 Two undirected graphs K_5, $K_{3,3}$ and a digraph K_3

Throughout this book the letter G always denotes a graph, which is directed or undirected according to the context if it is not specially noted.

A *system*, following Hayes [214], may be defined informally as a collection of objects, called components, connected to form a coherent entity with a well-defined function or purpose. The function performed by the system is determined by those performed by its components and by the manner in which the components are interconnected.

For a *computer system*, its components might include processors, control units, storage units and I/O (input/output) equipments (maybe include switches), and its function is to transform a set of input information items (e.g., a program and its

data set) into output information (e.g., the results computed by the program acting on the data set).

A *computer network* is a system whose components are autonomous computers and other devices that are connected together usually over long physical distance. Each computer has its own operating system and there is no direct cooperation between the computers in the execution of programs.

A *multiple processor system* (MPS) is a system whose components are two or more autonomous processors. Thus, an MPS may be thought of as an integrated computer system containing two or more processors. The qualification "integrated" implies that the processors cooperate in the execution of programs. MPS's consisting of thousands of processors are capable of executing parallel algorithms thus solving large problems in real time.

Following Saad and Schultz [401], there are essentially two broad classes of MPS architectures. The first class of MPS's is that its n identical processors are interconnected via a large switching network to n memories. The diagram of such a class of MPS architectures is shown in Figure 1.2. Variations on this scheme are numerous, but the essential feature here is the switching network. The main advantage of this type of configuration is that it enables us to make the data access transparent to the user who may regard data as being held in a large memory which is readily accessible to any processor. However, this type of memory-sharing architectures can not easily take advantage of some inherent properties in problems, for example, proximity of data where communication is local. Moreover, the switching network becomes exceedingly complex to build as the number of processors increases.

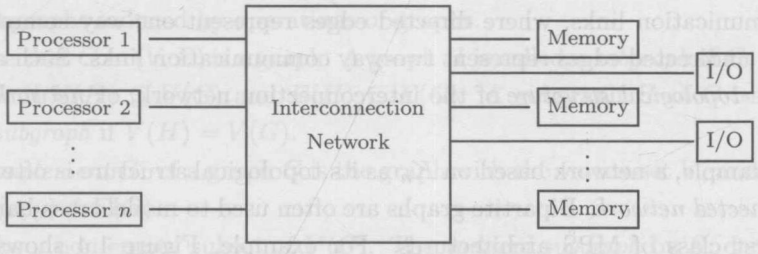


Figure 1.2 The first class of MPS architecture

The second important class of MPS architecture is that its processors, in which each processor has its own local memory, are interconnected according to some convenient pattern. The diagram of such a class of MPS architectures is shown in Figure 1.3. In this type of machine, there are no shared memory and no global synchronization. Moreover, intercommunication is achieved by message passing and computation is data driven. The main advantage of such architectures, often referred

to as ensemble architectures, is the simplicity of their design. The processors are identical, or are of a few different kinds, and can therefore be fabricated at relatively low cost.

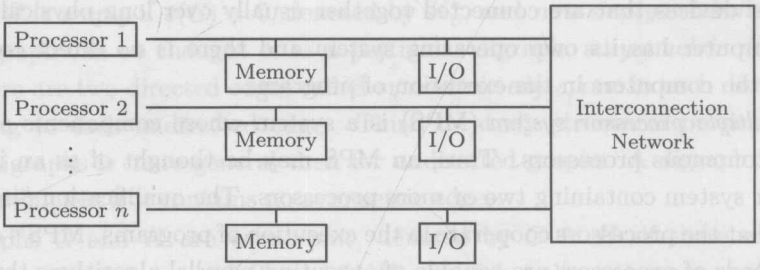


Figure 1.3 The second class of MPS architecture

A basic feature for a system is that its components are connected together by physical communication links to transmit information according to some pattern. Moreover, it is undoubted that the power of a system is highly dependent upon the connection pattern of components in the system.

A connection pattern of components in a system is called an *interconnection network*, or *network* for short, of the system. Topologically, an interconnection network can essentially depict structural feature of the system. In other words, an interconnection network of a system provides logically a specific way in which all components of the system are connected.

It is quite natural that an interconnection network may be modeled by a graph whose vertices represent components of the network and whose edges represent physical communication links, where directed edges represent one-way communication links and undirected edges represent two-way communication links. Such a graph is called the *topological structure* of the interconnection network, or *network topology* for short.

For example, a network based on K_n as its topological structure is often called a *fully connected network*. Bipartite graphs are often used to model cross-bar switches in the first class of MPS architectures. For example, Figure 1.4 shows a 3×3 cross-bar switch and its topological structure $K_{3,3}$.

Conversely, any a graph can also be considered as a topological structure of some interconnection network. Topologically, graphs and interconnection networks are the same things. Thus we will confuse a graph with a network. Instead of speaking a network, components, and links we speak of a graph, vertices and edges. The graph is directed or undirected, depending upon that the links are one-way or two-way in the network.

Usually the network topologies can be grouped into two categories: dynamic and

static. In a dynamic system such as the first class of MPS's mentioned above the links can be reconfigured by setting the network's active switching elements. In a static system such as the second class of MPS's the communication links between processors are passive and reconfiguration of the system is not possible. In this book, we are mainly interested in a static topological structure of interconnection networks.

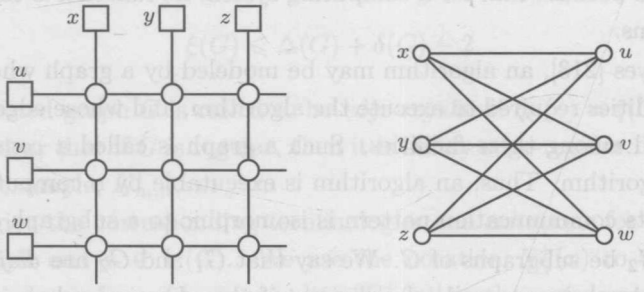


Figure 1.4 A 3×3 cross-bar switch and its topological structure

1.2 Basic Concepts and Notations

In this section, we will give some basic terminologies, notations and results on graphs used in this book, including subgraphs, degrees, paths, cycles, connected graphs, Euler circuits, Hamilton cycles, adjacency matrices, matchings, independence numbers, dominating numbers, and so forth.

A subgraph is one of the most basic concepts in graph theory. We first recall various subgraphs induced by operations of graphs.

Suppose that $G = (V, E)$ is a graph. A graph H is called a *subgraph* of G, denoted by $H \subseteq G$, if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. A subgraph H of G is called a *spanning subgraph* if $V(H) = V(G)$.

The *complement* G^c of a graph G is the graph with the vertex-set V, and $(x, y) \in E(G^c) \Leftrightarrow (x, y) \notin E(G)$.

Let S be a non-empty subset of $V(G)$. The *induced subgraph* by S, denoted by $G[S]$, is a subgraph of G whose vertex-set is S and whose edge-set is the set of those edges of G that have both end-vertices in S. The symbol $G - S$ denotes the induced subgraph $G[V \setminus S]$.

Let B be a non-empty subset of $E(G)$. The *edge-induced subgraph* by B, denoted by $G[B]$, is a subgraph of G whose vertex-set is the set of end-vertices of edges in B and whose edge-set is B. The notation $G - B$ denotes a subgraph of G obtained by deleting all edges in B. Similarly, the graph obtained from G by adding a set of edges F is denoted by $G + F$, where $F \subseteq E(G^c)$.

Subgraphs may be used to model a subnetwork of an interconnection network. If G is the topological structure of an interconnection network, then $G + F$ means addition of a set of links F to the network to improve its performance; $G - S$ and $G - B$ mean that the network contains a set S of faulty processors and a set B of faulty links, respectively.

It is essential that an interconnection network should contain some given kinds of subnetworks. It is because that for a computing system its function is the execution of some algorithms.

Following Hayes [213], an algorithm may be modeled by a graph whose vertices represent the facilities required to execute the algorithm, and whose edges represent the links required among these facilities. Such a graph is called a *communication pattern* of the algorithm. Thus, an algorithm is executable by a computing system G if and only if its communication pattern is isomorphic to a subgraph of G.

Let G_1 and G_2 be subgraphs of G. We say that G_1 and G_2 are *disjoint* if they have no vertices in common, and *edge-disjoint* if they have no edges in common. The *union* $G_1 \cup G_2$ of G_1 and G_2 is the subgraph with vertex-set $V(G_1) \cup V(G_2)$ and edge-set $E(G_1) \cup E(G_2)$; if G_1 and G_2 are disjoint, we sometimes denoted their union by $G_1 + G_2$. The *intersection* $G_1 \cap G_2$ of G_1 and G_2 is defined similarly if $V(G_1) \cap V(G_2) \neq \varnothing$.

The *join* $G \vee H$ of two disjoint undirected graphs G and H is the undirected graph obtained from $G + H$ by joining each vertex of G to each vertex of H.

We now recall the concepts related to degrees of a graph. We first consider undirected graphs. Let G be an undirected graph and $x \in V(G)$. We use the notation $E_G(x)$ to denote a set of edges incident with x in G. The cardinality $|E_G(x)|$ is called the *degree* of x, denoted by $d_G(x)$.

When an interconnection network is modeled by a graph G, the degree $d_G(x)$ of a vertex x in G corresponds the number of available connections to the component x in the network, which is bounded by the number of I/O devices attached to the component.

A vertex of degree d is called a *d-degree vertex*. 0-degree vertex is called an *isolated vertex*. A vertex is called to be *odd* or *even* if its degree is odd or even. A graph G is *d-regular* if $d_G(x) = d$ for each $x \in V(G)$, and G is *regular* if it is d-regular for some d, and d is the *regularity* of G. A graph G is *quasi-regular* if there are two distinct integers a and b such that $d_G(x) = a$ or b for any $x \in V(G)$. The parameters

$$\Delta(G) = \max\{d_G(x) : x \in V(G)\}, \quad \text{and}$$
$$\delta(G) = \min\{d_G(x) : x \in V(G)\}$$

are the *maximum* and *minimum degree* of G, respectively. For $xy \in E(G)$, the