UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

# RESISTANCE DISTANCE, KIRCHHOFF INDEX, FOSTER'S THEOREMS, AND GENERALIZATIONS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

MAMTA YADAV
Norman, Oklahoma
2017

RESISTANCE DISTANCE, KIRCHHOFF INDEX, FOSTER'S THEOREMS, AND GENERALIZATIONS

A DISSERTATION APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY

Dr. Krishnaiyan Thulasiraman, Chair

Dr. Sudarshan Dhall

Dr. Changwook Kim

Dr. Qi Cheng

Dr. Pramode Verma

Dr. Janet Allen

*To my spiritual guru SHREE SHREE MAA ANANTANAND TIRTH*

*To my parents, for encouraging me to follow my dreams*

# Acknowledgements

First, I would like to express my deepest gratitude to my advisor, Dr. Krishnaiyan Thulasiraman, who accepted me as his PhD student and offered me his mentorship, patience and care. This work would not have been possible without his excellent guidance and unconditional support. Under his guidance, I successfully overcame many difficulties and learned a lot. I am grateful for having the opportunity to work with him.

I would like to extend my thanks to members of my dissertation committee Dr. Sudarshan Dhall, Dr. Changwook Kim, Dr. Qi Cheng, Dr. Pramode Verma and Dr. Janet Allen for their valuable time, and invaluable feedback.

The School of Computer Science at OU has provided me with great learning experiences. I would like to thank Department Chair Dr. Sridhar Radhakrishnan, all the faculty and the wonderful staff, specially Virginie Perez, Lindsay Rice, Ashley Hill and James Casidy, for their guidance and help during all these years of graduate work.

I would wholeheartedly like to acknowledge the people who mean a lot to me, my spiritual Guru (Shree Shree Maa Anantanand Tirth) and my parents, Mr. Mohan Lal Yadav and Mrs. Shakuntala Yadav, for showing faith in me and giving me liberty to choose what I desired. I salute you all for the selfless love, care and sacrifice you did to shape my life.

In addition, I would like to acknowledge the love and support of my family: my uncle and aunty, Mr. Kalpesh Patel and Mrs. Surekha Patel; my sisters Manisha Yadav and Neetu Thakor; my brothers Dr. Shiv Yadav and Sanjeev Thakor; my brother-in-law Rohit Yadav; my sister-in-law Dr. Sudha Yadav; my little nieces Ramya Yadav and

Vishwa Thakor; and my little angel my daughter Durgey Yadav; for their support and generous care throughout this journey.

Finally, and most importantly, I owe thanks to a very special person, my husband Mr. Raman Yadav for his continued and unfailing love, support and understanding during my pursuit of PhD degree that made the completion of this dissertation possible.

# Contents

**Acknowledgements............................................................................................ iv**

**List of Figures.................................................................................................. ix**

**Abstract ............................................................................................................. xi**

**1 Introduction...........................................................................................1**

1.1 Introduction of Network Science.....................................................1
1.2 Why are we Interested in Networks.................................................3
1.3 A Brief History of Network Science................................................3
1.3.1 Early pre-network Period (1736 – 1966).......................................4
1.3.2 The meso-network Period (1967 – 1998)......................................5
1.3.3 The modern Period (1998 – present).............................................6
1.4 Key Aspects of Network Science ....................................................8
1.5 Networks .........................................................................................8
1.5.1 Definition of Network ....................................................................8
1.5.2 Types of Networks .......................................................................10
1.6 Overview of Resistance Distance and Kirchhoff Index..................13
1.7 Organization of the Dissertation.....................................................14

**2 Resistance Distance and Kirchhoff Index in Networks.....................................16**

2.1 Basic Definitions ...........................................................................17
2.2 Matrices of a Graph .......................................................................20
2.2.1 Incidence Matrix ..........................................................................20
2.2.2 Adjacency Matrix..........................................................................22
2.2.3 Laplacian Matrix...........................................................................23
2.2.4 Matrix-tree Theorem ....................................................................25
2.2.5 Pseudo-inverse of Laplacian Matrix .............................................27
2.3 Topological Formulas for Electrical Resistance Networks..............29
2.3.1 Resistance Networks ...................................................................29
2.3.2 Topological Formulas for Resistance Network Functions ............32
2.4 Kirchhoff Index of a Graph.............................................................40
2.4.1 Computation of the Kirchhoff Index using Laplacian Pseudo-inverse....40
2.4.2 A Simple Formula for the Kirchhoff Index Based on the Pseudo-inverse of the Laplacian Matrix ..............................................................42
2.5 Kirchhoff Index using Topological Formulas for Network Functions.............48
2.6 Kirchhoff Index using Circuit Theoretic Concepts ..........................49
2.7 Summary .......................................................................................51

**3 Cutset Laplacian Matrix of a Graph and Kirchhoff Index...............................53**

3.1 Cutsets...........................................................................................53
3.2 Cuts ...............................................................................................55
3.3 Fundamental Cutsets.....................................................................56
3.4 Cut Matrix and Fundamental Cutset Matrix ...................................59
3.5 Fundamental Circuit Matrix and Relationship with Fundamental Cutset Matrix..........................................................................................62

3.5.1 Fundamental Circuits ....62
3.5.2 Circuit Matrix ....62
3.5.3 Fundamental Circuit Matrix ....64
3.6 Kirchhoff's Laws and Fundamental Circuit and Cutset Matrices ....65
3.7 Cutset Laplacian Matrix and Kirchhoff Index ....66
3.8 Computing Kirchhoff Index: A Matrix Transformation Approach ....68
3.9 Kirchhoff Polynomial of a Graph and a Formula for Kirchhoff Index ....73
3.10 Summary ....79

**4 Weighted Kirchhoff Index of a Resistance Network and Generalization of Foster's Theorem ....80**
4.1 Basic Concepts and Definitions ....81
4.1.1 Star-Delta Transformation ....81
4.1.2 Multiple Star-Delta Transformations ....81
4.2 Foster's Theorems ....85
4.2.1 Foster's First Theorem ....85
4.2.2 Foster's Second Theorem ....86
4.3 Weighted Kirchhoff Index of a Resistance Network, Foster's Theorems, and Generalization ....89
4.3.1 Foster's First Theorem using Weighted Kirchhoff Index ....90
4.3.2 Foster's Second Theorem using Weighted Kirchhoff Index ....90
4.3.3 Generalized Foster's Theorem ....90
4.4 Dual Form of Foster's First Theorem ....93
4.5 Summary ....95

**5 Computing Kirchhoff Index ....96**
5.1 Introduction ....96
5.2 Graph Partition using METIS ....96
5.3 Graphics Processing Units (GPUs) and CUDA ....100
5.3.1 The Architecture of GPU ....100
5.3.2 CUDA Programming Model and Memory Model ....101
5.4 Star-Delta Transformation Algorithm using Series and Parallel Reduction ....103
5.5 Sequential and Parallel Approaches for Resistance Distance Computation ....105
5.5.1 Sequential Approach for Resistance Distance ....105
5.5.2 Parallel Approach for Resistance Distance ....108
5.6 Three Step Approximation Algorithm for Resistance Distance Calculation 115
5.7 Experiment Results ....119
5.8 Summary ....121

**6 Conclusions ....122**
6.1 Summary ....122
6.2 Future Directions of Research ....123
6.2.1 Graph Clustering ....124
6.2.2 Similarity and Criticality Measures ....125

**Bibliography ....128**

# List of Tables

Table 1: Key Aspects of Network Science ....................................................................9

Table 2: Types of Networks ..........................................................................................12

Table 3: Comparison of time for sequential and parallel implementation ...................120

Table 4: Comparison of time for number of processors for graph G250 .....................120

Table 5: Quality performance of Three Step Approximation Algorithm.....................120

# List of Figures

Figure 1.1 History of Network Science ........................................................4

Figure 1.2 Königsberg bridge problem.........................................................5

Figure 2.1 Effective Resistance using series - parallel method and Star-Delta Transformation ........................................................................18

Figure 2.2 Incidence Matrix.........................................................................21

Figure 2.3 A weighted directed graph G and its all-vertex incidence matrix.............25

Figure 2.4 Spanning 3-tree...........................................................................27

Figure 2.5 A network element.......................................................................29

Figure 2.6 Directed graph representation of an electrical network............................30

Figure 2.7 A 1-port network. ........................................................................33

Figure 2.8 A 2-port network. ........................................................................35

Figure 2.9 Laplacian Matrix .........................................................................46

Figure 2.10 A 2-port Network .......................................................................50

Figure 2.11 Three node network. ...................................................................50

Figure 3.1 Illustration of the definition of a cutset...................................................54

Figure 3.2 A set of fundamental cutsets of a graph..................................................57

Figure 3.3 Spanning tree of directed graph.............................................................61

Figure 3.4 Set of two fundamental circuits of $G$......................................................63

Figure 3.5 Voltage $Vj$ across the $j$th branch when a current source of $1A$ is..............68

Figure 3.6 Star tree $Tn$.........................................................................................69

Figure 3.7 Voltages across the branches and current injected through branches. .......69

Figure 3.8 Spanning tree T and star tree.................................................................71

Figure 3.9 Path from vertex i to j. ...........................................................................74

Figure 3.10 3-node equivalent representation of the graph given in Figure 3.9...........75

Figure 3.11 ...........................................................................................................75

Figure 3.12 ...........................................................................................................76

Figure 3.13 ...........................................................................................................77

Figure 4.1 Star – delta transformation........................................................................82
Figure 4.2 Multiple Star-Delta Transformation ..........................................................82
Figure 4.3 A 5-node resistance network N. ..................................................................83
Figure 4.4 Corresponding network N(D) for each subset D of nodes..........................84
Figure 4.5 Calculating effective resistance distance between vertices $v1$ and $v2$.......88

Figure 5.1 Graph partition files. ....................................................................................98
Figure 5.2 Graph G partitioned in four clusters. ...........................................................99
Figure 5.3 GPU Architecture (NVIDIA Tesla C1060). ..............................................101
Figure 5.4 CUDA Programming Model (Corporation, 2010) .....................................102
Figure 5.5 CUDA Memory Model (Corporation, 2010). ...........................................103
Figure 5.6 Illustration of Star-Delta Transformation and series/parallel reductions..104
Figure 5.7 Illustration of Star-Delta Transformation Procedure Algorithm. .............106
Figure 5.8 Adjacency Matrix *A* of graph *G* given in Figure 5.1(a)...........................113
Figure 5.9 Resistance Distance Matrix *R* of graph *G* given in Figure 5.1(a)……….114
Figure 5.10 Boundary node network *B′* of graph *G*…………………………………117
Figure 5.11 Resistance distance matrix *R* using three step approximation…………118

# Abstract

The emerging area of network science studies structural characteristics of networks and dynamical processes on networks such as spread of epidemics, vulnerability of power grids to cascading failures etc. In this area, several measures of network performance have been introduced and studied. In this dissertation, we study two measures, namely, resistance distance and Kirchhoff index.

Treating each element of a graph as a resistance, resistance distance between two nodes *u* and *v* is the effective resistance across *u* and *v*. Kirchhoff index defined by the chemistry community is the sum of the effective resistances across all pairs of nodes of the graph. Kirchhoff index, also called network criticality, has been studied by the communication network community. Kirchhoff index has been studied using the graph Laplacian matrix which is the same as the indefinite admittance matrix of a resistance network.

Our research is on reducing the computational effort in calculating the Kirchhoff index in networks. First a simpler formula for Kirchhoff index based on the properties of node-to-datum resistance matrix is presented. To avoid computational complexity and extraneous efforts of Moore-Penrose pseudoinverse, Kirchhoff index is calculated in terms of the inverse of the reduced Laplacian matrix.

The notion of Laplacian matrix is then generalized using the fundamental cutset matrix of a graph. Two approaches to compute Kirchhoff index are presented: The first approach is based on a matrix transformation, and the second approach uses the concept of Kirchhoff polynomial of a graph. Kirchhoff polynomial of a graph introduced in this work is defined for each spanning tree of the graph.

In 1949 and 1961 Foster established two theorems that give identities involving resistance distances. We introduce the concept of Weighted Kirchhoff index of a graph and study its relationship to Foster’s theorems. We present a generalization of Foster’s theorems that retains the circuit-theoretic flavor and elegance of Foster’s theorems, and develop a dual form of this theorem.

Kirchhoff index captures the effect of topological structure on the performance of networks. It also captures the path diversity between nodes in a network. Kirchhoff index can be used to determine node betweenness in networks that are of interest in network vulnerability studies. In view of this, an efficient methodology to compute Kirchhoff index is required. For this purpose, we propose sequential and parallel algorithms. In addition, we introduce a novel 3-step approximation algorithm for calculation of resistance distance and Kirchhoff index.

# Chapter 1

# Introduction

## 1.1 Introduction to Network Science

Complex systems are pervasive in our society. Some examples are the Internet System that interconnects computer networks globally, the World Wide Web System that links the information networks to each other, the electrical power system, the biological system that relates the networks of biologically relevant entities, the communication system that integrates billions of cell phones with satellites and computers, the social system that interrelate the individuals, groups, institutions, organizations etc. There are three aspects to study the complex systems. The first is the study of the nature of the individual components of the systems, the second is the study of the nature of connections or interactions and the third is the study of the *pattern* of connections between components.

Networks represent the pattern of connections in a system. The science of networks is called network science. This is not a new concept, and it has roots as far back as 1736. Network science has roots in many subfields, for example, in social network analysis, electrical circuits and systems, synthetic emergent systems (i.e. the Internet, power grid), biological science etc.

Network science is defined in many ways by the National Research Council (NRC) of the National Academies. The most direct definition given by NRC is (Lewis, 2009):

**Definition 1.1.** *Network science is an organized knowledge of networks based on their study using the scientific methods.*

In simple language, a network is a collection of points that are joined together by lines. Each subfield has a different working definition of a network. For communication engineers the network is a system of routers and switches and for marketing business people it is a population of buyers. According to sociologists a network is an influence diagram that represents the social interaction among humans and for physicists it is a model of phase transition and magnetism. Biologists use network analogy to understand the epidemics and metabolic system within a cell but for power engineers a network is a system of electrical power grids.

The operational definition of network science has two main components (Newman, 2010):

(i) Network science is the study of the structure of a collection of nodes and links that represent something real.

(ii) Network science is the study of the dynamic behavior of the aggregation of nodes and links.

The nodes might be molecules or genes for biological systems, humans for social systems, routers or switches for communication systems, transformers for electrical systems. The links might be contagions or synapses for biological system, friendships or other relationships for social systems, physical wires or wireless for communication systems, cables for electrical systems, etc.

## 1.2 Why are we Interested in Networks

To understand complex systems, we have to acquire a deep understanding of the networks behind the systems. A network reduces a complex system to an abstract structure representing the connection patterns in the system. A network can be described by a graph structure (i.e. nodes and links) and by its behavior (i.e. the interaction among the nodes and links). Over the years, scientists have developed a pervasive set of mathematical, computational, and statistical tools for analyzing, modeling and understanding networks. These tools work with networks in their abstract form and help in finding some crucial and useful information about networks, for example, the critical node or edge in a network, length of a path from one node to another in a network, flow of traffic over the network, clusters or communities in a network, etc. These tools can be applied to any systems that can be represented as networks.

## 1.3 A Brief History of Network Science

Network science is not only a single field, but it is a result of convergence of many other subfields. The two major evolutions in network science are: (i) from mathematical theory to graph theory and (ii) from graph theory to collections of generalization about the things that are connected.

The history of network science can be divided into three periods (Newman, 2010) as shown in Figure 1.1.

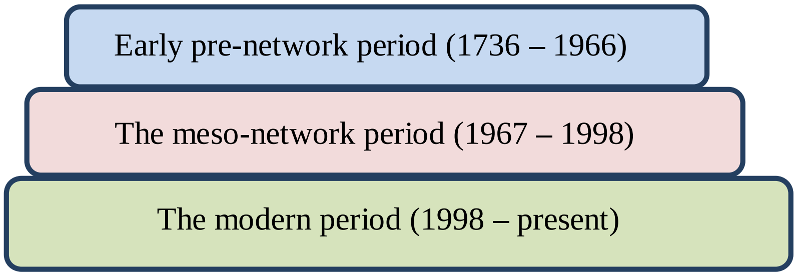

Early pre-network period (1736 – 1966)
The meso-network period (1967 – 1998)
The modern period (1998 – present)

Figure 1.1: History of network science

### 1.3.1 Early pre-network period (1736 – 1966)

Early pre-network period is the period when network science was really the mathematics of graphs. The very first known application of network science was Euler’s treatment of Bridges of Königsberg (Euler, 1736). This application established graph theory and demonstrated that many real-world problems can be solved by abstractions as graphs. Euler called a graph a mathematical object consisting of points (or nodes) and lines (or edges). In his study, Euler represented the four land masses as four vertices and joined them by seven edges in the pattern of the Königsberg bridges (Figure 1. 2). The problem is to start at one vertex, traverse all the edges exactly once and return to the starting vertex.

In network science, the next major turning point was in 1925, when Yule first observed *preferential attachment* in evolution (Yule, 1925). Preferential attachment describes an emergent process and it explains the existence of scale-free networks in natural and synthetic systems. In 1927, Kermack and McKendrick discovered a mathematical epidemic model of the spread of a disease in biological networks. Their idea of epidemics was first applied by Solomonoff and Rappaport in 1951 to random networks.

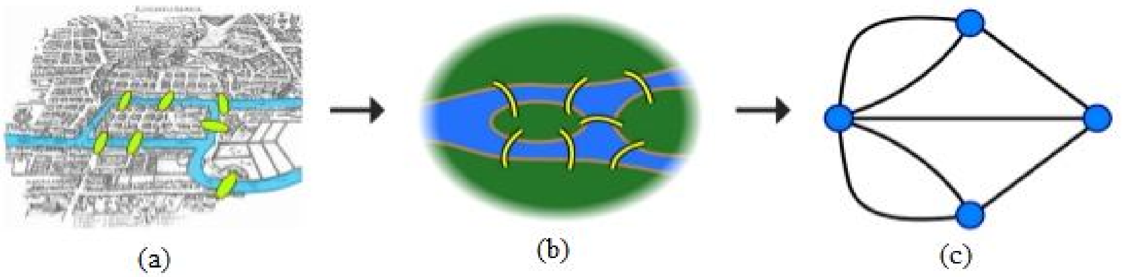


Figure 1.2 (Anon., 2003): Königsberg bridge problem. (a) A map of eighteenth century Königsberg with its seven bridges. (b) Simplified illustration of the rivers and bridges in the Königsberg bridge problem. (c) the corresponding network of nodes and edges.

By the mid-twentieth century, network science figured out that the nature and real objects could be modeled as random processes or as random graphs. In 1959, Gilbert built a random graph in two steps, the first step was to construct a complete graph and the second step was to delete the randomly selected links from the graph until it reached the desired number of links (Gilbert, 1959). But very soon in 1960, Erdos and Renyi, surpassed Gilbert's algorithm and came up with an elegant and simple algorithm which is widely used today. Erods-Renyi (ER) algorithm constructs a network of *n* nodes by inserting a link between randomly selected pair of nodes and this process is repeated until *m* links have been inserted (Erdos & Renyi, 1960). By late 1960s the seed of network science was planted in seemingly unrelated disciplines.

### 1.3.2 The meso-network period (1967 – 1998)

This is the period when applications of networks started emerging. In 1967, a major turning point in network science was marked by Stanley Milgram by his "six degrees of separation" experiment. Milgram called this network *a small-world network* because he concludes that the social world is smaller than the real world and it took only six hops to connect a pair of strangers, regardless of where they lived. Milgram's small-world idea is based on the "weak ties" theory. Later in 1973, Granoveter (Granovetter, 1973) gave his theory that social networks contain both "strong ties" and "weak ties". *Strong ties* are the direct connection between two nodes and *weak ties* are the long-distance connections that bind social world. In 1978, Pool and Kochen determined the theoretical analysis of small-world networks. Bonacich was the first social scientist who postulated the mathematical representation of the social networks by using the connection matrix (Bonacich, 1972). The Marketing gurus remark that the highly-connected people are *superspreaders*, while on the other hand the social scientists note that the middle-person or intermediary person is powerful and called it *betweenesss*. Betweenness is the number of paths that must run through a node to connect to other nodes.

Kuramoto's work in 1984 on synchronization in coupled linear systems has had a major impact on convergence between network science and control theory (Kuramoto, 1984). The fundamentals of network science had been established by 1998. This was the time when Internet was at rapid rise and Waxman proposed a static graph theory model of Internet (Waxman, 1988).

### 1.3.3 The modern period (1998 – present)

*Emergence* plays a very crucial role in the study of networks. In 1998, Holland defined emergence as "a major change in global properties of networks coming from small changes at the local level" (Holland, 1998). Watts and Strognatz showed their interest in small-world networks and generated arbitrarily small world networks that fall between a random network and non-random network (Watts & Strogatz, 1998; Watts, 1999; Watts, 1999a). After this, the small world networks were not restricted to social networks only. The year 1999 turns out to be a milestone for the modern period, as this year was full of discoveries. M. Faloutsos, P. Faloutos and C. Faloutos observed a power law in their Internet graph model (Faloutsos, et al., 1999), and similarly Albert, Jeong, and Barbasi observed power law in their WWW model (Albert, et al., 1999). In (Barbasi, et al., 1999) Barbasi and Albert determined a generative procedure to produce *scale-free networks*.

Dorogovstsev, Mendes, Samukhim, Krapivsky, and Redner introduced the concept of power law of purely scale-free networks in many biological systems (Dorogovtsev, et al., 2000; Dorogovtsev, et al., 2002; Dorogovtsev & Mendes, 2002; Dorogovtsev & Mendes, 2003). In 2000, Kleinberg showed that it takes *O (n)* steps to search a small world using “Manhattan distance” (Kleinberg, 2000). Albert, Jeong, and Barbasi observed that the scale-free networks are resilient for *protected hubs* (Albert, et al., 2000).

Wang, Chen, Barahona, Pecora, Liu, Hong, Choi, Jost, Joy and others showed the stability of any network as a function of the network’s topology (Wang & Chen, 2002; Wang & Chen, 2002a; Wang & Chen, 2002b; Barahona & Pecora, 2002; Liu, et al., 2002; Liu, 2003; Liu, et al., 2004; Liu, et al., 2004a; Hong & Kim, 2002; Jost & Joy, 2002).

Wang, Chakrabarti, Wang, and Faloutsos determined the spread of epidemics by using the largest eigenvalue of connection matrix and network’s spectral radius (Wang, et al., 2003).

Atay et al. (Atay, et al., 2006) studied synchronization in networks with the degree sequence distribution. Lewis (Lewis, 2009) extended the topological results of networks to several classes of Atay's network and to a new class of networks called *Kirchhoff newtorks*. Atay's network uses a local averaging algorithm to compute the state of nodes (Atay, et al., 2006), while the new class of Kirchhoff Networks stabilizes the value of nodes by maintaining the Kirchhoff's first law. Recently network science has contributed to many results in many fields such as marketing, electrical engineering, biology, communication systems, etc.

## 1.4 Key Aspects of Network Science

To investigate the topology and dynamics of several systems, network science uses different tools such as graph theory, social network analysis, market competition modeling, epidemic modeling, etc. Network science is distinguished by the subject of study as well as by its methodology. Some key aspects of Network Science are given in Table 1.

## 1.5 Networks

### 1.5.1 Definition of network

In simplest form, a network is a collection of points joined together in pairs by lines. The points are called *nodes* or *vertices* and the lines are called *links* or *edges*.

A complete definition of network must include both structural and behavioral information (Lewis, 2009).

| **Aspects** | **Description** |
|---|---|
| *Structure* | Networks are not just a random collection of nodes and links, but networks have structure. For example, social networks are not just a collection of people connected randomly, but instead, the networks have a distinct format or topology.  The nodes of a network, unite in a distinct format to form a structure. |
| *Topology* | The pattern in which the nodes of a network are connected is called topology. In dynamic networks, the topology changes as a function of time. Topology is a consequence of Darwinian forces that shape the network. |
| *Emergence* | Network science is the study of both static and dynamic properties of networks. The emergent property helps a dynamic network in achieving stability. Emergence is a network synchronization issue. A dynamic network transits from one state to another state until either cycling back or reaching a fixed point. The evolution of a network from initial state to future state is a called emergence. |
| *Power* | The power of a node is proportional to its degree i.e., the number of links connecting to the network of the power of a network is proportional to the strength of its nodes and links. |
| *Stability* | A network is dynamically stable if the rate of change in the state of its topology diminishes as time passes. |
| *Bottom-up evaluation* | Networks evolve from local level to the global level. They are designed and implemented by using bottom-up strategy. |

Table 1: Key aspects of network science

The structural information of a network is modeled by the corresponding graph. The behavioral information about networks is defined by a set of *microrules* governing the behavior of nodes and links.

**Definition 1.5.1:** *For a given network G,*

$$G(t) = \{V(t), E(t), f(t): J(t)\},$$

where,

$G(t)$ is a function of time *t*

*t* = time, simulated or real

*V* = nodes or vertices

*E* = links or Edges

$f: N \times N$ = mapping function that connects nodepairs, yielding topology

*J* = “ *microrules*” or algorithm for describing behaviors of nodes and links versus time.

### 1.5.2 Types of networks

Networks are divided into four general classes (Newman, 2010):

(i) Technological networks

(ii) Social networks

(iii) Information networks

(iv) Biological networks

The list of some of the most important examples in each class and their description is given in Table 2.

| Classes | Examples | Nodes and Edges | Description |
|---|---|---|---|
| *Technological Networks* | The Internet | *Nodes*: Computers or other devices<br>*Edges*: wires or wireless | The Internet is a network of physical data connections between computers and related devices. |
| | The Telephone network | *Nodes*: Telephones or mobile phones<br>*Edges*: Wires or wireless | The telephone network is a network of landlines and wireless links that transmit telephone calls. |
| | Power Grids | *Nodes*: Generating stations and switching substations<br>*Edges*: High-voltage lines | A power grid is a network of high-voltage transmission lines that provide long-distance transport of electric power within and between countries. |
| | Transportation Networks | *Nodes*: Geographic locations<br>*Edges*: Routes between geographic locations | Transportation networks describe the flow of some commodity or vehicular movement between geographic locations. Some examples of transportation networks are airline route networks, road networks and rail networks. |
| *Social Networks* | Facebook, Twitter, MySpace. | *Nodes*: People or groups of people<br>*Edges*: social interaction, such as friendship. | A social network is a network of people (such as friends, coworkers) connected by some social relationships (such as friendship). Sociologists call vertices (or people) as *actors* and the edges as *ties*. |
| *Information Networks* | The World Wide Web | *Nodes*: Web pages consisting of text, pictures or other information. | The world-wide web is a network of web pages that are connected to each other by means of hyperlinks. Hyperlinks |

|  |  |  |  |
|---|---|---|---|
|  |  | *Edges*: hyperlinks or hypertexts. | allow us to navigate from one web page to another. |
|  | Citation Networks | *Nodes*: papers<br>*Edges*: citation | In citation networks, there is a direct edge from paper A to paper B if paper A cites paper B in its bibliography. |
| *Biological Networks* | Biochemical networks | *Nodes*: molecules (genes, proteins, metabolites, cells etc.)<br>*Edges*: interaction (reactions, molecular interaction, regulatory interaction etc.) | Biochemical networks represent the molecular level patterns of interaction and mechanisms of control in the biological cell. Examples of Biological networks are metabolic networks, protein-protein interaction networks, and genetic regulatory networks. |
|  | Neural networks | *Nodes*: neurons<br>*Edges*: excitatory inputs, inhibiting inputs. | A neural network is a network that models the brain and central nervous system in animals. The neurons are connected by two types of directed edges, one for excitatory inputs and one for inhibiting inputs. |
|  | Ecological Networks | *Nodes*: species, individuals<br>*Edges*: interaction between species. | Ecological network is a network of ecological interactions between species. Examples of ecological networks are Food web networks, host-parasite networks, mutualistic networks, etc. |

Table 2: Types of networks

## 1.6 Overview of Resistance Distance and Kirchhoff Index

As discussed in the previous sections, graphs and networks have been used extensively in many applications (Newman, 2010; Easley & Kleinberg, 2010; Barabasi, 2013; Chiang, 2012). In these works, several network measures have been defined and studied. Of these measures, closeness and betweenness measures of nodes and edges that capture their criticality have received a great deal of attention. In defining these measures, paths between nodes play an important role. Though, in general, all paths must be used in assessing the centrality of a node, shortest paths are used because they are easy to compute. To mitigate the effect of the approximation of criticality by considering only shortest paths, other measures that capture both the lengths of paths and the number of these paths between nodes need to be investigated. Resistance distance and Kirchhoff Index are two such exemplary measures. To capture accurately the impact of paths, resistance distance can be used in place of shortest distances and Kirchhoff index can be used in place of the sum of all shortest distances. This motivates our study in this dissertation

Resistance distance is based on the electrical network theory and it was first introduced by Klein and Randić (Klein & Randic, 1993). The concept of resistance distance has been much studied in the chemical studies (Klein & Randic, 1993; Xiao & Gutman, 2003). Resistance distance implies many dynamic properties of a graph or network. The properties of resistance distances were proved using the Laplacian matrix (Xiao & Gutman, 2003; Xiao & Gutman, 2003a). Resistance distance and Kirchhoff index have wide applications in complex networks, chemistry, physics, electric circuit, graph theory and others. The concept of the Kirchhoff analysis was first introduced by G. Kirchhoff (Kirchhoff, 1847) in 1847 for the graph-theoretic study of electric circuits.

Resistance distance across a pair of nodes is the same as the effective resistance across that pair, treating each edge as a 1 ohm resistance. A special case of this restricted to only the edges of a graph was studied by Foster (Foster, 1949). A further generalization of this was given by Foster in (Foster, 1961). In (Tetali, 1994) Tetali proved Foster’s first theorem using certain results from the theory of Markov chains, then Palacios gave an extension of Foster’s second theorem in (Palacios, 2004). Generalization of all of the Foster’s theorems are given by Cinkir in (Cinkir, 2011). The connection between resistance distance and random walks on graphs have been discussed in (Thulasiraman, et al., 2015; Doyle & Snell, 1984).

Kirchhoff index is the sum of the resistance distances across all pairs of nodes in the network. Kirchhoff index has also been studied using the graph Laplacian. The Laplacian of a graph is the same as the indefinite admittance matrix of a resistance network that has been studied by electrical circuit theorists extensively in the development of several results (Swamy & Thulasiraman, 1981). See (Molitierno, 2012) for detailed study of the Laplacian from a graph-theoretic perspective.

## 1.7 Organization of the Dissertation

The rest of the dissertation is organized as follows. Chapter 2 discusses the relationship between resistance distance and Kirchhoff index. A new formula for Kirchhoff index is presented in this chapter. The generalization of Laplacian matrix using the fundamental cutset matrix is introduced in Chapter 3. Two approaches to compute Kirchhoff index are presented in this chapter. Chapter 4 generalizes the notion of Kirchhoff index and studies its relationship to Foster’s theorems. A dual form of Foster’s first theorem is developed in this chapter. We propose sequential and parallel algorithms for resistance distance in Chapter 5. A novel approximation algorithm for resistance distance and Kirchhoff index is introduced in this chapter. Conclusion and future work is given in Chapter 6.

# Chapter 2

## Resistance Distance and Kirchhoff Index in Networks

Over the past several years a variety of graph measures have been proposed to reveal the behavior of networks based on topological and dynamical characteristics. Resistance distance and Kirchhoff index are highly valuable graph measures in the study of various network problems. These measures were first studied in the chemical literature. In recent years, they have also attracted the attention of researchers in electrical engineering, mathematics, computer science and social networks.

In 1993, Klein and Randić (Klein & Randic, 1993) introduced the concept of resistance distance. The resistance distance concept is the convergence of resistive electrical network theory and the graph theory. An electrical resistance network can be viewed as a connected graph, with the junctions in the electrical network as the vertices of the graph and the unit resistors of one ohm as the edges of the graph. The effective resistance between pairs of vertices is called the resistance distance between these vertices. Kirchhoff index of a graph is the sum of resistance distances between all pairs of vertices. The Laplacian matrix of a graph plays an important role in the computation of resistance distance and Kirchhoff index. The standard method to obtain resistance distance is via Moore-Penrose pseudoinverse $L^+$ of the Laplacian matrix L of a connected graph *G* (Klein & Randic, 1993; Zhu, et al., 1996).

This chapter is concerned with the study of relationship between resistance distance and Kirchhoff index. In the following section, we briefly present certain basic definitions in graph theory. For other graph theory concepts not covered in section 2.1 (Kirchhoff, 1847) may be consulted.

## 2.1 Basic Definitions

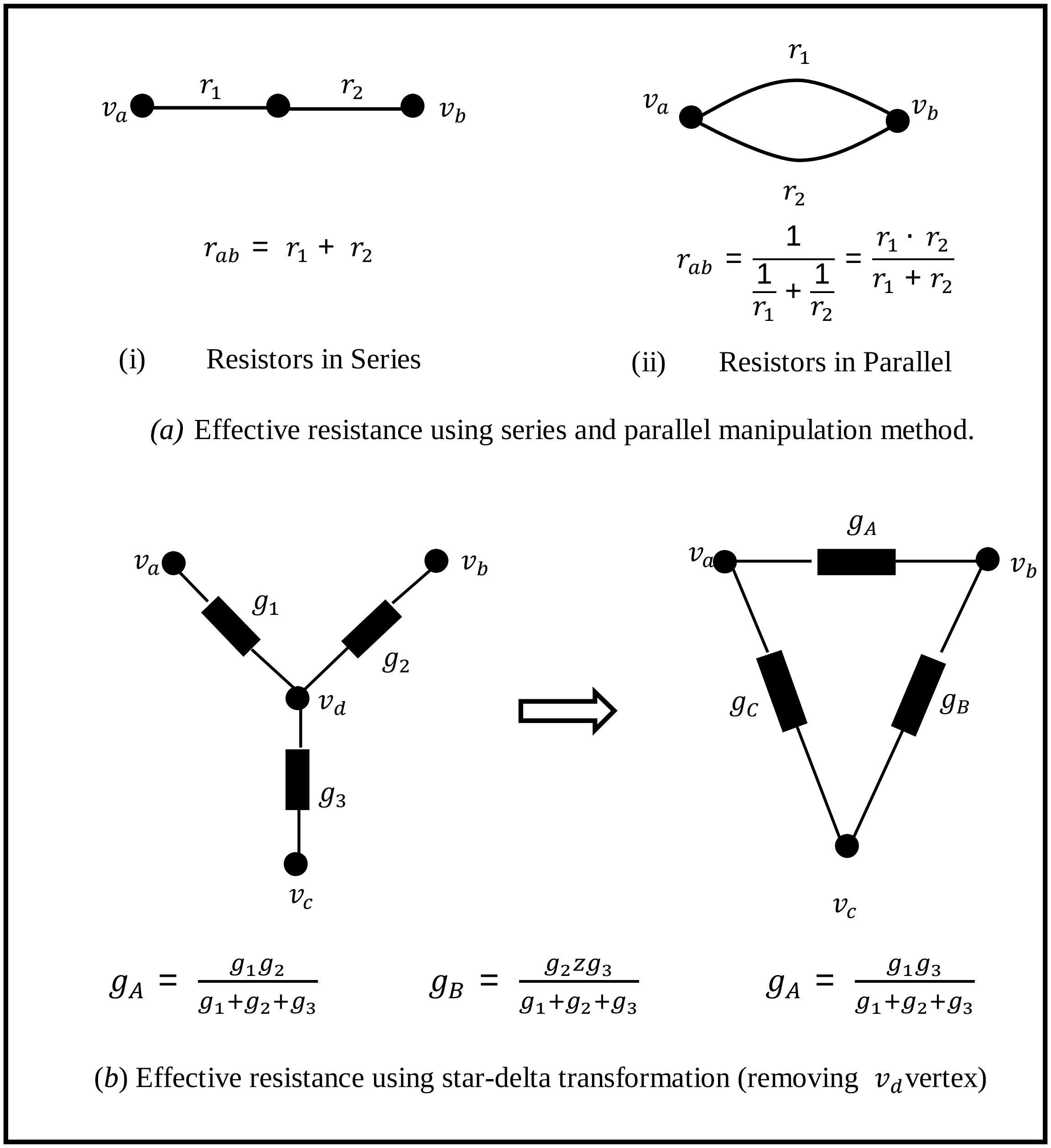
Let $G = (V(G), E(G))$ be a connected graph with the vertex set $V(G) = \{v_1, v_2, v_3, \ldots\ldots, v_n\}$ and the edge set $V(G) = \{e_1, e_2, e_3, \ldots\ldots, e_m\}$, where $n$ is the number of vertices and $m$ is the number of edges. Let $N$ be an electrical network obtained from the connected graph $G$. To obtain an electrical network from the graph $G$, replace each edge of $G$ with a unit resistor.

The resistance distance $r_{ij}$ between vertices $v_i$ and $v_j$ of graph $G$ is defined as the effective resistance between vertices $v_i$ and $v_j$ of the electrical network $N$. The effective resistance $r_{ij}$ is the potential difference between vertices $v_i$ and $v_j$ when unit current is injected into $v_i$ and drawn from $v_j$.

The effective resistance between two vertices of an electrical circuit can easily be calculated by the well-known series and parallel manipulation and star-delta transformation. Figure 2.1 (a) illustrates the series and parallel manipulation method to calculate the effective resistance distance $r_{ab}$ between vertices $v_a$ and $v_b$. Figure 2.1 (b) illustrates the start-delta transformation to calculate the effective resistance distance by using conductance $g_i$ , which is the reciprocal of conductance $g_i$ , i.e., $r_i = \frac{1}{g_i}$ .

The Kirchhoff index is a structure descriptor (Xiao & Gutman, 2003a) based on the resistance distance. The Kirchhoff index $Kf(G)$ of the graph G is defined as

$$Kf(G) = \sum_{i<j} r_{ij} \ . \qquad (2.1)$$



$$r_{ab} = r_1 + r_2$$

$$r_{ab} = \frac{1}{\frac{1}{r_1} + \frac{1}{r_2}} = \frac{r_1 \cdot r_2}{r_1 + r_2}$$

(i) Resistors in Series

(ii) Resistors in Parallel

*(a)* Effective resistance using series and parallel manipulation method.

$$g_A = \frac{g_1 g_2}{g_1+g_2+g_3} \qquad g_B = \frac{g_2 z g_3}{g_1+g_2+g_3} \qquad g_A = \frac{g_1 g_3}{g_1+g_2+g_3}$$

($b$) Effective resistance using star-delta transformation (removing $v_d$vertex)

Figure 2.1: Effective resistance using (a) series-parallel method (b) star-delta transformation.

The resistance distance and Kirchhoff index have been extensively studied in chemical literature. Kirchhoff index appears in several applications: electrical networks, Markov chain, averaging networks, and experiment design (Klein & Randic, 1993; Kirchhoff, 1847; Bonchev, et al., 1994; Hu, et al., 2013; Hu, et al., 2013a). The formula for Kirchhoff index has been computed for some classes of graphs such as cycle-containing graphs (Klein, et al., 1995; Lukovits, et al., 1999), complete graphs (Lukovits, et al., 1999), circulant graphs (Zhang & Yang, 2007) and distance transitive graphs (Palacios, 2001). Bapat (Bapat, 2004) obtained a formula for the inverse and determinant of resistance distance for weighted graphs by using the properties of resistance distance and Kirchhoff index defined by Xiao and Gutman (Xiao & Gutman, 2003a). Several properties of the Kirchhoff index related to the normalized Laplacian eigen values of a connected graph are presented by Zhou and Trinajstic (Zhou & Trinajstic, 2009).

In 1993, Kunz (Kunz, 1993) studied the properties of the Laplacian matrix for finding the topological distances in the graph. In 1949, Foster (Foster, 1949) discussed the concept of the effective resistance distance and recently in 2004 this concept was again studied by Palacios (Palacios, 2004). Palacios used effective resistance distance to extend the Foster's first and second formulas and then used Foster's third formula to compute the Kirchhoff index of a class of graphs with diameter 3. Further review of literature on Foster's theorems will be given in Chapter 4.

In this chapter, we study the relationship between resistance distance, Kirchhoff index and the Laplacian matrix of a graph. Section 2.2 discusses the incidence, adjacency and Laplacian matrices of a graph, Section 2.3 discusses the topological formulas for resistance network functions. Section 2.4 describes the basic facts and notations of Laplacian graph spectral theory. A new formula for the Kirchhoff index of a graph is presented in section 2.4. Three proofs of this formula based on the properties of the pseudo-inverse of the Laplacian matrix, topological formula for network functions and basic concepts of electrical circuit theory are presented.

## 2.2 Matrices of a Graph

In this section, we introduce the incidence, adjacency and Laplacian matrices of a graph and establish several properties of these matrices that help to reveal the structure of a graph (Swamy & Thulasiraman, 1981). The incidence, adjacency and Laplacian matrices arise in the study of electrical network because these matrices are the coefficient matrices of the Kirchhoff's equation that describes a network. Thus, the properties of these matrices form the basis of graph-theoretic study of electrical networks and systems, in particular, resistance distance and Kirchhoff index.

### 2.2.1 Incidence matrix

Consider a graph *G* with *n* vertices and *m* edges and having no self-loops. The *all-vertex incidence matrix* $A_c = [a_{ij}]$ of *G* has *n* rows, one for each vertex, and *m* columns, one for each edge. The element $a_{ij}$ of $A_c$ is defined as follows:

*G* is *undirected*

$$a_{ij} = \begin{cases} 1, & \text{If the } j \text{the edge is incident on the } i \text{the vertex;} \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

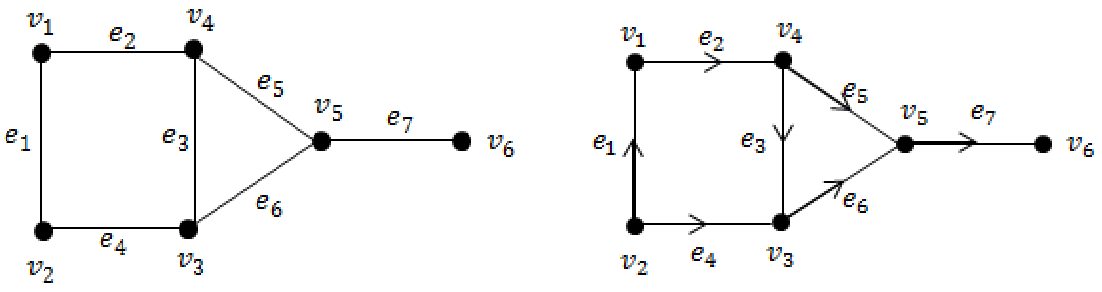
*G* is *directed*

$$a_{ij} = \begin{cases} 1, & \text{if the } j\text{th edge is incident on the } i\text{th vertex and} \\ & \text{oriented away from it;} \\ -1, & \text{if the } j\text{th edge is incident on the } i\text{th vertex and} \\ & \text{oriented toward it ;} \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

A row of $A_c$ will be referred to as an *incidence vector* of $G$. Two graphs and their all-vertex incidence matrices are shown in Figures 2.2*a* and 2.2*b*.

It should be clear from the preceding definition that each column of $A_c$ contains exactly two non-zero entries, one +1 and one -1. Therefore, we can obtain any row of $A_c$ from the remaining $n-1$ rows. Thus, any $n-1$ rows of $A_c$ contain all the information about $A_c$ . In other words the rows of $A_c$ are lineraly dependent.

An ($n$ −1)-rowed submatrix $A$ of $A_c$ will be referred to as an incidence matrix of $G$. The vertex which corresponds to the row of $A_c$ which is not in $A$ will be called the reference vertex or datum vertex of $A$.



$$A_c = \begin{array}{c|ccccccc} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \\ \hline v_1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v_2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v_3 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ v_4 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ v_6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

$$A_c = \begin{array}{c|ccccccc} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \\ \hline v_1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ v_2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ v_3 & 0 & 0 & -1 & -1 & 0 & 1 & 0 \\ v_4 & 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 0 & -1 & -1 & 1 \\ v_6 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{array}$$

(*a*) (*b*)

Figure 2.2: Incidence matrix. (*a*) An undirected graph *G* and its all-vertex incidence matrix. (*b*) A directed graph *G* and it’s all vertex incidence matrix.

Note that

$$\text{rank}(A) = \text{rank}(A_c) \leq n - 1$$

In the case of a connected graph, the rank of $A_c$ is in fact equal to $n - 1$. This result is based on the following theorem.

**Theorem 2.1** *The determinant of any incidence matrix of a tree is equal to* ±1.

See (Swamy & Thulasiraman, 1981) for a proof of the above theorem.

Since a connected graph has at least one spanning tree, it follows from Theorem 2.1 that in any incidence matrix *A* of a connected graph with *n* vertices there exists a nonsingular submatrix of order $n - 1$. Thus, for a connected graph *A*,

$$\text{rank}(A) = n - 1.$$

Since $\text{rank}(A_c) = \text{rank}(A)$, we get the following theorem.

**Theorem 2.2.** *The rank of the all-vertex incidence matrix of an n-vertex connected graph G is equal to n-1, the rank of G.*

An immediate consequence of Theorem 2.2 is the following.

**Corollary 2.2.1.** *If an n-vertex graph has p components, then the rank of its all-vertex incidence matrix is equal to n – p, the rank of G.*

### 2.2.2 Adjacency matrix

Let $G = (V, E)$ be a directed graph with no parallel edges. Let $V = \{v_1, v_2, \cdots, v_n\}$. The *adjacency matrix* $M = [m_{ij}]$ of *G* is an $n \times n$ matrix with $m_{ij}$ define as follows:

$$m_{ij} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E. \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

In the case of an undirected graph, $m_{ij}$ = 1 only if there is an edge connecting $v_i$ and $v_j$. For example, the undirected graph of Figure 2.2($a$) has the following adjacency matrix:

$$M = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

and the directed graph of Figure 2.2($b$) has the following adjacency matrix:

$$M = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Clearly, for undirected graphs, the adjacency matrix $M$ is a symmetric matrix with zeros on the diagonal.

### 2.2.3 Laplacian matrix

Let $G = (V, E)$ be a weigthed graph with vertex set $V(G) = \{v_1, v_2, \ldots, v_n\}$ and edge set $E(G)$. Let $w_{ij}$ denote the weight of edge $(i, j)$. The adjacency matrix $M(G)$ is as defined in (2.4). Then the degree matrix $D(G)$ is defined as

$$D_{i,j} = \begin{cases} \text{sum of the weights of the edges incident on } i & if\ i = j \\ 0 & otherwise \end{cases} \qquad (2.5)$$

Note that if each $w_{ij} = 1$, then $D_{i,i}$ is equal to the degree of $i$.

The Laplacian matrix of a weighted graph $G$ is a square matrix of order $n$, defined by

$$L(G) = D(G) - M(G)\,. \qquad (2.6)$$

Note that, the $(i, j)$- entry of Laplacian matrix $L$ can be written as:

$$L_{i,j} = \begin{cases} -w_{ij} & if\ i \neq j\ and\ v_i\ and\ v_j\ are\ adjacent \\ 0 & if\ i \neq j\ and\ v_i\ and\ v_j\ are\ not\ adjacent \\ \text{Sum of the weights of the} & if\ \ i = j \\ \text{edges incidnet on } i & \end{cases} \quad (2.7)$$

So $L = A_c W A_c^t$ where $W$ is the diagonal matrix with the diagonal entries representing the weights on the edges.

Let $L\overline{(\imath)}$ be a reduced Laplacian matrix which is obtained by removing *i*th row and *i*th column from $L$. The reduced Laplacian matrix of a graph *G* is given by

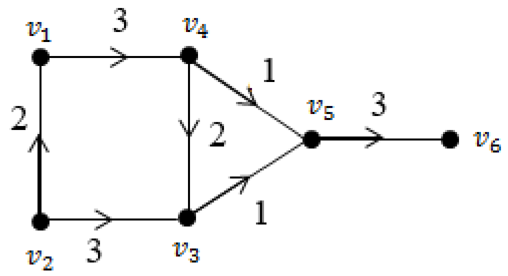
$$L\overline{(\imath)} = AWA^t \quad (2.8)$$

The Laplacian matrix and reduced Laplacian matrix of a weighted directed graph *G* (Figure 2.3) is calculated as follows. The diagonal matrix *W* for given graph is

$$W = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}.$$

We can now calculate Laplacian matrix by using $L = A_c W A_c^t$.

$$L = \begin{bmatrix} 5 & -2 & 0 & -3 & 0 & 0 \\ -2 & 5 & -3 & 0 & 0 & 0 \\ 0 & -3 & 6 & -2 & -1 & 0 \\ -3 & 0 & -2 & 6 & -1 & 0 \\ 0 & 0 & -1 & -1 & 5 & -3 \\ 0 & 0 & 0 & 0 & -3 & 3 \end{bmatrix}$$



$$A_c = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{array} \\ \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \end{array}$$

Figure 2.3: A weighted directed graph G and its all-vertex incidence matrix.

The reduced Laplacian matrix after deleting the $n^{th}$ row of the incidence matrix $A_c$ using (2.8) is

$$L(\bar{\iota}) = \begin{bmatrix} 5 & -2 & 0 & -3 & 0 \\ -2 & 5 & -3 & 0 & 0 \\ 0 & -3 & 6 & -2 & -1 \\ -3 & 0 & -2 & 6 & -1 \\ 0 & 0 & -1 & -1 & 5 \end{bmatrix}.$$

### 2.2.4 Matrix-tree theorem

#### 2.2.4.1 The number of spanning trees

A spanning tree of a graph *G* is a tree of *G* having all the vertices of *G*. The spanning trees of a connected graph are in one-to-one correspondence with the nonsingular submatrices of matrix *A*.

**Theorem 2.3.** *A square submatrix of order $n-1$ of any incidence matrix A of an n-vertex connected graph G is nonsingular if and only if the edges that correspond to the columns of the submatrix form a spanning tree of G.*

Given a spanning tree of a graph *G*, the product of all the weights of edges in the spanning tree is called the tree weight product. We denote by $\tau(G)$ the sum of the weights product of all spanning tree of *G*.

**Theorem 2.4.** *Let G be a connected and weighted undirected graph and A be an incidence matrix of the directed graph that is obtained by assigning arbitrary orientations the edges of G. Then*

$$\tau(G) = det(AWA^t) = det\,L(\bar{\iota})\,, \qquad for\ any\ vertex\ i.$$

Thus, from Theorem 2.4 we get the following result, originally due to Kirchhoff (Kirchhoff, 1847).

**Theorem 2.5.** *All the cofactors of the degree matrix of a connected undirected graph has the same value as the number of spanning trees of G.*

### 2.2.4.2 The number of spanning 2-trees

A *k*-tree is an acyclic graph consisting of *k* components. If a *k*-tree is a spanning subgraph of a graph *G*, then it is called a *spanning k-tree* of *G*. The spanning 2-trees $T_{ijk\dots,rst\dots}$ denotes a 2-tree, in which the vertices $v_i, v_j, v_k, \dots$ are required in one component and the vertices $v_r, v_{s,} v_t, \dots$ are required to be in the other component of the 2-tree. For example, Figure 2.4(b) shows an example of a spanning 3-tree of the graph *G* shown in Figure 2.4($a$).

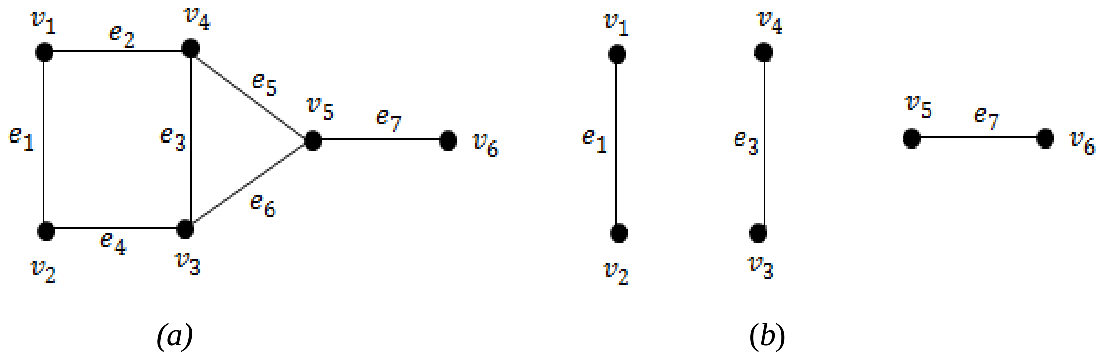


Figure 2.4: Spanning tree. (*a*) Graph *G* (*b*) A spanning 3-tree *T* of *G*.

The sum of weight products of all spanning 2-trees of type $T_{ij\ldots,rs\ldots}$ will be denoted by $\tau_{ijk\ldots,rst\ldots}$. Let $\Delta_{ij}$ denote the $(i, j)$ cofactor of $AWA^t$. That is, $\Delta_{ij}$ is the $(i,j)$ cofactor of $L(\bar{k})$ for any $k$.

**Theorem 2.6.** *For a connected graph G,*

$$\Delta_{ii} = \tau_{i,n} \text{ and}$$

$$\Delta_{ij} = \tau_{ij,n}$$

### 2.2.5 Pseudo-inverse of Laplacian matrix

The sum of elements in each row and the sum of elements in each column of a Laplacian matrix is zero, that is,

$$\sum_{i=1}^{n} L_{ij} = \sum_{j=1}^{n} L_{ij} = 0\,. \qquad (2.9)$$

So, the determinant of Laplacian matrix is zero, that is, $\det L(G) = 0$.

Since the determinant of the Laplacian matrix is zero, it has no inverse. So, the Moore-Penrose pseudoinverse of $L(G)$ is used as a substitute for the inverse of $L(G)$.

The Moore-Penrose pseudoinverse of Laplacian matrix $L(G)$ is denoted by $L^{+}(G)$ and has the following basic properties

(i) $L(G)L^{+}(G)L(G) = L(G)$

(ii) $L^{+}(G)L(G)L^{+}(G) = L^{+}(G)$

(iii) $[L(G)L^{+}(G)]' = L(G)L^{+}(G)$

(iv) $[L^{+}(G)L(G)]' = L^{+}(G)L(G)$

The Moore-Penrose pseudoinverse $L^{+}(G)$ can be computed as follows (Gutman & Mohar, 1996):

$$L^{+}(G) = \left(L(G) + \frac{J}{n}\right)^{-1} - \frac{J}{n} \qquad (2.10)$$

where $J \in R^{nxn}$ is a matrix of all 1's and *n* is the number of vertices of graph $G$.

The following properties were established and proved by several authors (Gutman & Xiao, 2004) for the Moore-Penrose pseudoinverse of the Laplacian matrix.

**Lemma 2.7** (Klein, et al., 1995)**.** *The Moore-Penrose pseudoinverse $L^{+}(G)$ of the Laplacian matrix $L(G)$ of a connected graph is real and symmetric.*

**Lemma 2.8** (Klein, et al., 1995)**.** *The Laplacian matrix and its pseudoinverse satisfy the following relations*

$$L(G)J = JL(G) = 0$$

$$L^{+}(G)J = JL^{+}(G) = 0$$

**Lemma 2.9** (Klein, et al., 1995)**.** *If $L(G)$ and $L^{+}(G)$ pertain to a connected graph on n vertices, then*

$$L(G)L^{+}(G) = L^{+}(G)L(G) = I - \frac{J}{n}$$

**Theorem 2.10** (Klein, et al., 1995)**.** *If $G$ is a connected graph, then the inverse of the matrix $L(G) + \frac{J}{n}$ exists and is equal to $L^{+}(G) + \frac{J}{n}$.*

*Proof.* Using Lemma 2.8 and Lemma 2.9, and the fact that $J^2 = nJ$, we have

$$\left(L(G) + \frac{J}{n}\right)\left(L^{+}(G) + \frac{J}{n}\right) = L(G)L^{+}(G) + \frac{J}{n}L^{+}(G) + \frac{1}{n}L(G)J + \frac{1}{n^2}J^2$$

$$= \left(I - \frac{J}{n}\right) + O + O + \frac{J}{n} = I.$$ □

## 2.3 Topological Formulas for Electrical Resistance Networks

### 2.3.1 Resistance networks

An electrical network is an interconnection of electrical network elements such as resistances, capacitances, inductances and voltage and current sources. We will assume that all the network elements in the networks to be considered are resistances. Each network element is associated with two variables, the voltage variable $v(t)$ and the current variable $i(t)$. We need to specify reference directions for these variables because they are functions of time and may take on positive and negative values in the course of time. This is done by assigning an arrow, called orientation, to each network element (Figure 2.5). This arrow means that $i(t)$ is positive whenever the current is in the direction of the arrow. Further we assume that the positive polarity of the voltage $v(t)$ is at the tail end of the arrow. Thus $v(t)$ is positive whenever the voltage drop in a network element is in the direction of the arrow.

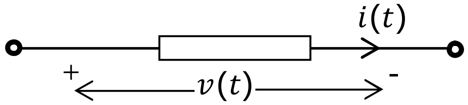


Figure 2.5: A network element (representation).

Network elements are characterized by the physical relationships between the associated voltage and current variables. Ohm’s law specifies the relationship between $v(t)$ and $i(t)$ as

$$v(t) = R\, i(t) \tag{2.11}$$

where $R$ is the resistance (in ohms) of the network element.

Note that for some of the network elements the voltage variables may be required to have specified values and for some others the current variables may be specified. Such elements are called, respectively, the voltage and current sources.

Two fundamental laws of network theory are Kirchhoff’s laws, that are stated as follows:

**Kirchhoff’s Current Law (KCL):** *The algebraic sum of the currents flowing out of a vertex is equal to zero.*

**Kirchhoff’s Voltage Law (KVL):** *The algebraic sum of the voltages around any circuit is equal to zero.*

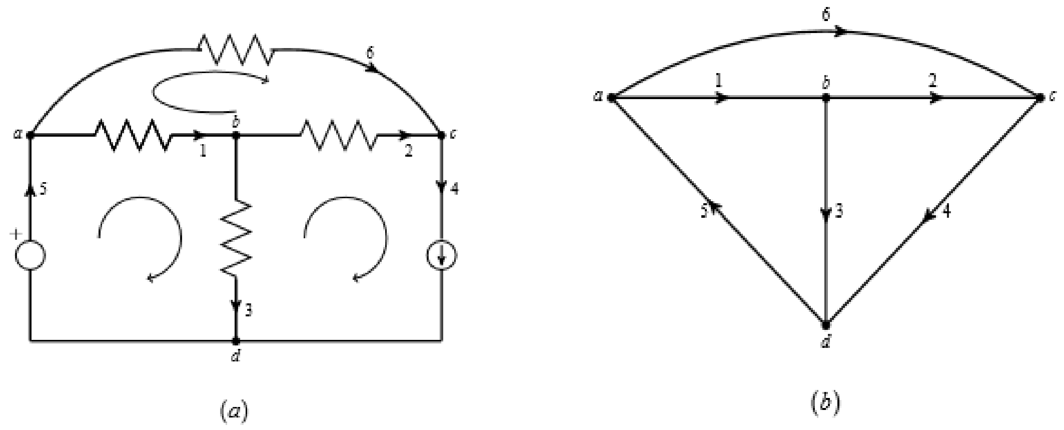


Figure 2.6: Directed graph representation of an electrical network. (*a*) Electrical Network *G*. (*b*) Directed graph of *G*

For instance, for the network shown in Figure 2.6($a$) the KCL and KVL equations are as given below. In this figure element 5 is a voltage source and element 4 is a current source.

*KCL equations*:

vertex $a$ = $i_1 - i_5 + i_6 = 0$,

vertex $c$ = $-i_2 + i_4 - i_6 = 0$,

vertex $b$ = $-i_1 + i_2 + i_3 = 0$.

*KVL equations*:

circuit {1, 3, 5} $v1 + v3 + v5 = 0$

circuit {2, 4, 3} $v2 + v4 - v3 = 0$

circuit {1, 6, 2} $-v1 + v6 - v2 = 0$

Given an electrical network $G$, the problem of network analysis is to determine the element voltages and currents that satisfy Kirchhoff's laws and the Ohm's law.

Notice that the equations which arise from an application of Kirchhoff's laws are algebraic in nature, and they depend only on the way the network elements are interconnected and not on the nature of the network elements. There are several properties of an electrical network which depend on the structure of the network. In studying such properties, it will be convenient to treat each network element as a directed edge associated with the two variables $v(t)$ and $i(t)$ . Thus, we may consider an electrical network as a directed graph in which each edge is associated with the two variables $v(t)$ and $i(t)$, which are required to satisfy Kirchhoff's laws and the Ohm's law. For example, the directed graph corresponding to the network of Figure 2.6(a) is shown in Figure 2.6($b$).

It is now easy to see that KCL and KVL equations for a network $G$ can be written, respectively, as

$$Q_c I_e = 0 \tag{2.12}$$

and

$$B_c V_e = 0 \tag{2.13}$$

where $Q_c$and $B_c$ are the cut and circuit matrices of the directed graph associated with $G$, and $I_e$ and $V_e$ are, respectively, the column vectors of element currents and voltages of *N*.

Since the all-vertex incidence matrix $A_c$ is a submatrix of $Q_c$ and has the same rank as $Q_c$, we can use in equation (2.3) the matrix $A_c$ in place of $Q_c$. Thus, KCL equations can be written as

$$A_c I_e = 0 \tag{2.14}$$

Since the rank of $A_c$ is $n-1$, we can remove any row from $A_c$ and use the resulting matrix *A* called the incidence matrix. The vertex corresponding to the removed row is called the reference or datum vertex.

### 2.3.2 Topological formulas for resistance network functions

Consider first a 1-port resistance network *G*. Each port is defined by a pair of nodes. The network is available for connection through the ports to the other parts of a system. Let the network *G* have $n + 1$ nodes denoted by 0, 1, 2, ..., *n*, and let the nodes 1 and 0 be, respectively, the positive and negative reference terminals of the port (Figure 2.7).

Let us now excite the network by connecting a current source of value $I_1$ across the port. Let $V_1$, $V_1$ , $V_2, \ldots , V_n$ denote the voltages of the nodes 1, 2, ..., $n$ with respect to node 0. This means $V_0$ = 0 and $V_i$ is the voltage between the nodes $i$ and 0 (that is $V_i = V_i - V_0$ ) for $i \neq 0$. Also, the $A$ matrix does not contain the row corresponding to the vertex 0.

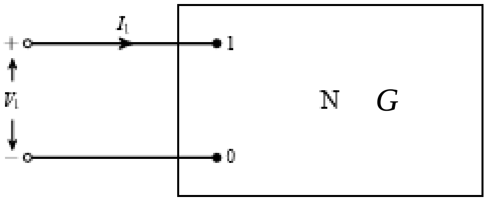


Figure 2.7: A 1-port network.

Then we have

$$A\, I_e - I = 0,$$

that is

$$A\, I_e = I \qquad (2.15)$$

where,

$$I = \begin{bmatrix} I_1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Note that in the graph representation of a port, the corresponding edge will be oriented from the positive terminal to the negative terminal. So, the current flowing through this in the direction of the orientation is $-I_1$ where the voltage from positive terminal to negative terminal of the port is $v_1$.

Let the network elements be labeled as $e_1, e_2, \ldots, e_m$ with $r_i$ denoting the resistance value of element $e_i$. Then the conductance of $e_i$ is given by $w_i = \frac{1}{r_i}$. Let $W$ be the diagonal matrix with its ($i$, $i$) entry equal to $w_i$. Then we can write

$$I_e = WV_e \tag{2.16}$$

Suppose the end vertices of $e_i$ are $k$ and $l$. Then the voltage across this element (voltage drop from node $k$ to node $l$) is given by $V_k - V_l$ , assuming that the element is oriented from vertex $k$ to vertex $l$. So, we can write

$$V_e = A^t V \tag{2.17}$$

where $V$ is the vector of voltages $V_1, V_2, \ldots, V_n$. Combining (2.15), (2.16) and (2.17) we get the node equations

$$AWA^t V = I \tag{2.18}$$

where

$$V = \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ \vdots \\ V_n \end{bmatrix}$$

Let

$$Y = AWA^t$$

so, that

$$Y\,V = I \tag{2.19}$$

Note that the matrix $Y$ is the same as the reduced Laplacian $L(\bar{0})$ defined in section 2.2.3.

The matrix $Y$ is called the node-conductance matrix of the network with vertex 0 as the reference. Solving (2.19) for $V_1$, we get

$$V_1 = \frac{\Delta_{11}}{\Delta} I_1 ,$$

where

$$\Delta = \det Y$$

and

$$\Delta_{11} = (1,1) \text{ cofactor of } Y.$$

So, the driving-point resistance across vertices 1 and 0 is given by

$$z = \frac{V_1}{I_1} = \frac{\Delta_{11}}{\Delta} , \tag{2.20}$$

and the driving-point conductance across 1 and 0 is given by

$$y = \frac{1}{z} = \frac{\Delta}{\Delta_{11}} \quad . \tag{2.21}$$

To illustrate certain principles of network analysis, consider next a 2-port network *G* (See Figure 2.8). If the ports of *G* are excited by current sources of values $I_1$ and $I_2$, then the node equations of *G* can be written as

$$YV = I$$

where

$$I = \begin{bmatrix} I_1 \\ I_2 \\ -I_2 \\ \vdots \\ 0 \end{bmatrix} .$$

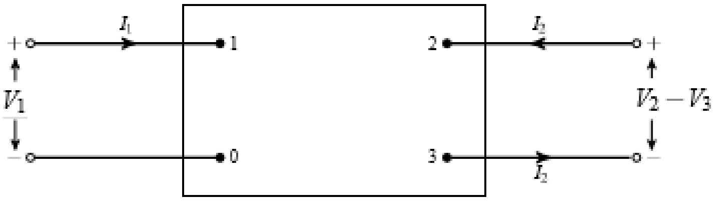


Figure 2.8: A 2-port network.

Solving for the node voltages $V_1$, $V_2$, and $V_3$, we get

$$V_1 = \frac{1}{\Delta}(\Delta_{11}I_1 + \Delta_{21}I_2 - \Delta_{31}I_2),$$

$$V_2 = \frac{1}{\Delta}(\Delta_{21}I_1 + \Delta_{22}I_2 - \Delta_{32}I_2),$$

$$V_3 = \frac{1}{\Delta}(\Delta_{31}I_1 + \Delta_{31}I_2 - \Delta_{33}I_2),$$

From the above relations, we get

$$\begin{bmatrix} V_1 \\ V_2 - V_3 \end{bmatrix} = \frac{1}{\Delta}\begin{bmatrix} \Delta_{11} & \Delta_{21} - \Delta_{31} \\ \Delta_{12} - \Delta_{213} & \Delta_{22} + \Delta_{33} - \Delta_{32} - \Delta_{23} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \tag{2.22}$$

$$= Z_{oc}I$$

Here $Z_{oc}$ is called the open circuit resistance matrix of the 2-port network. This is because each element of $Z_{oc}$ is obtained by setting one of the port currents equal to zero (that is, open-circuiting the corresponding port). Thus

$$z_{11} = \frac{V_1}{I_1}\bigg|\ I_2 = 0,$$

$$z_{12} = \frac{V_1}{I_2}\bigg|\ I_1 = 0,$$

$$z_{21} = \frac{V_2}{I_1}\bigg|\ I_2 = 0,$$

$$z_{22} = \frac{V_2}{I_2}\bigg|\ I_1 = 0,$$

Here $z_{11}$ and $z_{22}$ are called driving point resistances across the respective ports and $z_{12}$ and $z_{21}$ are called transfer resistances between the ports. Note that since $Y$ is symmetric, we have

$$\Delta_{ij} = \Delta_{ji}.$$

So

$$Z_{oc} = \frac{1}{\Delta}\begin{bmatrix} \Delta_{11} & \Delta_{21} - \Delta_{31} \\ \Delta_{12} - \Delta_{213} & \Delta_{22} + \Delta_{33} - \Delta_{32} - \Delta_{23} \end{bmatrix}. \qquad (2.23)$$

Thus, from Theorem 2.5, we have the following results

$$\Delta = \tau(G)$$

$$\Delta_{11} = \tau_{1,0}\ . \qquad (2.24)$$

Recall that $\tau(G)$ is the sum of the conductance products of all the spanning trees in $G$ and $\tau_{1,0}$ is the sum of the conductance products of all the spanning 2-trees of the type $T_{1,0}$ (with 1 and 0 in separate trees of $T_{1,0}$). So

$$\Delta_{ij} = \tau_{ij,0}$$

where $\tau_{ij,0}$ is the sum of the conductance products of all 2-trees $T_{ij,0}$ ($i$ and $j$ in one tree and 0 in the other tree). So

$$\Delta_{12} - \Delta_{13} = \tau_{12,0} - \tau_{13,0}\ . \qquad (2.25)$$

Since each spanning 2-tree $T_{12,0}$ is either a spanning 2-tree $T_{12,30}$ or a spanning 2-tree $T_{123,0}$, we get

$$\tau_{12,0} = \tau_{12,30} + \tau_{123,0}\ . \qquad (2.26)$$

Similarly,

$$\tau_{13,0} = \tau_{13,20} + \tau_{123,0}\ . \qquad (2.27)$$

Then

$$\Delta_{12} - \Delta_{13} = \tau_{12,0} - \tau_{13,0}\ . \qquad (2.28)$$

By a similar reasoning,

$$\begin{aligned}\Delta_{22} + \Delta_{33} - 2\Delta_{23} &= \tau_{2,0} + \tau_{3,0} - 2\tau_{23,0} \\ &= \tau_{23,0} + \tau_{2,30} + \tau_{23,0} + \tau_{3,20} - 2\tau_{23,0} \\ &= \tau_{2,30} + \tau_{3,20} \\ &= \tau_{2,3} \qquad (2.29)\end{aligned}$$

So, we can write $Z_{oc}$ as

$$Z_{oc} = \frac{1}{\tau(G)}\begin{bmatrix} \tau_{1,0} & \tau_{12,30} - \tau_{13,20} \\ \tau_{12,30} - \tau_{13,20} & \tau_{2,3} \end{bmatrix}.$$

So, the driving point resistance across port 1 is given by

$$Z_{1,0} = \frac{\tau_{1,0}}{\tau(G)}$$

Similarly, the driving point resistance $z_2$ across 2 and 3 is given by $\frac{\tau_{2,3}}{\tau(G)}$ . In general, the driving point resistance across any pair of nodes *i* and *j* is given by $\frac{\tau_{i,j}}{\tau(G)}$. We shall denote by $r_{ij}$ the driving point resistance across any pair of vertices *i* and *j* so that

$$r_{ij} = \frac{\tau_{i,j}}{\tau(G)} \qquad (2.30)$$

where $r_{ij}$ is also called the effective resistance across *i* and *j*.

We wish to emphasize that the formulas for $z_{ij}$'s in (2.13) are with respect to vertex 0 as reference. On the other hand, the formula in (2.22) does not explicitly involve the reference vertex. We conclude this subsection with the following facts that will be needed in the subsequent sections, where we shall assume that the vertices are labeled as $1, 2, \cdots, n$

1. The degree matrix $K = [K_{ij}]$ of a simple undirected graph $G = (V, E)$ is defined as

   $$k_{ii} = d(v_i), \qquad \text{for all } i \in V$$

   $$k_{ij} = -1, \qquad \text{if } (i,j) \in E$$

   $$= 0 \qquad \text{otherwise}$$

   where $d(v_i)$ is the degree of vertex *i*. Then *K* can be written as

   $$K = A_c A_c^t$$

   where $A_c$ is the all-vertex incidence matrix of *G*.

2. Let $N$ be the resistance network $N$ obtained by associating a 1-ohm resistance with each edge of *G*. Then in electrical engineering literature the matrix *K* is called the indefinite conductance matrix. In graph theory literature *K* is also known as the graph Laplacian. Also, if the conductances are defined by $g_i$, with *G* as the diagonal matrix of edge conductances, then the graph Laplacian of the corresponding weighted graph will be $A_c G A_c^t$. Here the degree of vertex *i* is the sum of the conductances incident on *i*.

3. Let $K_{jj}$ be the matrix obtained by removing the *j*th row and the *j*th column from *K*. Then $K_{jj}$ is the same as the matrix *Y* defined in (2.18) with vertex *j* as reference if all the resistances have 1 ohm value.

4. By Theorem 2.5 all cofactors of *K* are equal to the number of spanning trees of *N*. In particular

   $$\det K_{jj} = W. \tag{2.31}$$

5. The (*i*,*i*) cofactor of $K_{jj}$ = number of spanning 2-trees of the type

   $$T_{i,j} = W_{i,j}. \tag{2.32}$$

6. The (*i*,*k*) cofactors of $K_{jj}$ = Number of spanning 2-trees of the type

$$T_{ik,j} = W_{ik,j}. \qquad (2.33)$$

7. The effective resistance $r_{ij}$ across *i* and *j* of *N* is given by

$$r_{ij} = \frac{(i, i\ ) \text{ cofactor of } Y}{\text{determinent } Y}$$

$$= \frac{(i, i\ ) \text{ cofactor of } K_{jj}}{\text{determinent } K_{jj}}$$

$$= \frac{(i, i\ ) \text{ cofactor of } K_{jj}}{W}$$

$$= \frac{W_{i,j}}{W}. \qquad (2.34)$$

## 2.4 Kirchhoff Index of a Graph

The structural and functional robustness of a network can be measured by the Kirchhoff index. The Kirchhoff index $Kf(G)$ of a connected undirected graph *G* is defined as

$$Kf(G) = \sum_{i<j} r_{ij}. \qquad (2.35)$$

Thus $Kf(G)$ is the sum of the effective resistances across all pairs of vertices of the 1-ohm resistance network obtained from *G*.

### 2.4.1 Computation of the Kirchhoff index using Laplacian pseudo-inverse

In a network, the resistance distance $r_{i,j}$ between any pair of nodes $i$ and $j$ can be computed by using the Kirchhoff Law and Ohm law. The Moore-Penrose pseudoinverse $L^{+}(G)$ gives the following formula (Klein & Randic, 1993; Xiao & Gutman, 2003) for computing the resistance distance $r_{i,j}$:

$$r_{ij} = L_{ii}^{+} + L_{jj}^{+} - L_{ij}^{+} - L_{ji}^{+}. \qquad (2.36)$$

From Lemma 2.7 we know that the Moore-Penrose pseudoinverse is symmetric. So now the equation (2.36) can be simplified as

$$r_{ij} = L^+_{i,i} + L^+_{jj} - 2L^+_{ij}.$$

Kirchhoff index $Kf(G)$ is the sum of the resistance distance of all pair of vertices of a graph $G$:

$$Kf(G) = \sum_{i<j} r_{ij} = \sum_{i<j}\left(L^+_{i,i} + L^+_{jj} - 2L^+_{ij}\right)$$

It was proved by Klein and Randic (Klein & Randic, 1993) that the Kirchhoff Index can also be written as

$$Kf(G) = ntr\left(L^+(G)\right) \tag{2.37}$$

where $n$ is the number of vertices and $tr(L^+(G))$ denotes the trace function which can be calculated by

$$tr\left(L^+(G)\right) = \sum_{i=1}^{n} l^+_{ii}.$$

Gutman and Mohar (Gutman & Mohar, 1996) demonstrated that it is possible to calculate the Kirchhoff Index without knowing the Moore-Penrose pseudoinverse of a Laplacian matrix. They obtained the Kirchhoff Index from the eigenvalues of the Laplacian matrix of a graph $G$:

$$Kf(G) = n\sum_{i=1}^{n-1} \frac{1}{\mu_i} \tag{2.38}$$

where $\mu_i$ is the non-zero eigenvalues of the Laplacian matrix $L(G)$.

To avoid the computational efforts required to calculate the Moore-Penrose pseudoinverse of the Laplacian matrix, we next present a new formula for $Kf\,(G)$.

### 2.4.2 A Simple formula for the Kirchhoff index based on the pseudo-inverse of the Laplacian matrix

Let $L$ be the Laplacian matrix of a connected graph $G$ and $L(\bar{\imath})$ be a submatrix obtained by deleting the $i^{th}$ row and $i^{th}$ column of the Laplacian matrix $L$. Note that $L(\bar{\imath})$ is the same as the node-conductance $Y$, if vertex $i$ is chosen as reference.

Let $Z$ be the inverse of $L(\bar{\imath})$, i.e.,

$$Z = L(\bar{\imath})^{-1}. \tag{2.39}$$

**Theorem 2.11** (Molitierno, 2012)**.** *Let $L$ be the Laplacian matrix of a connected graph $G$ with $n$ vertices. Then*

$$L^{+} = \frac{e^T Le}{n^2} J + \left[\begin{array}{c|c} Z - \frac{1}{n}ZJ - \frac{1}{n}JZ & -\frac{1}{n}Ze \\ \hline -\frac{1}{n}e^T Z & O \end{array}\right] \tag{2.40}$$

*where $e$ is the left and right null vector of any Laplacian matrix and matrix $Z$ is the inverse of a reduced Laplacian matrix obtained by deleting the last ($n^{th}$) row and the last ($n^{th}$) column, that is, $Z = L(\bar{n})^{-1} = Y^{-1}$.*

*Proof.* By Lemma 2.9 we know that

$$L^{+}L = LL^{+} = I - \frac{J}{n}$$

where, $L$ is the Laplacian matrix, $L^{+}$ is the pseudoinverse of Laplacian matrix $L$, $I$ is the identity matrix, $J$ is a unit matrix of all 1's and $n$ is the number of vertices of graph $G$.

Multiply $L$ on both sides of equation (2.40):

$$L^{+}L = \left(\frac{e^{T}Le}{n^{2}}J + \left[\begin{array}{c|c} Z - \frac{1}{n}ZJ - \frac{1}{n}JZ & -\frac{1}{n}ze \\ \hline -\frac{1}{n}e^{T}Z & O \end{array}\right]\right)L$$

$$= \frac{e^{T}Le}{n^{2}}JL + \left[\begin{array}{c|c} Z - \frac{1}{n}ZJ - \frac{1}{n}JZ & -\frac{1}{n}ze \\ \hline -\frac{1}{n}e^{T}Z & O \end{array}\right]L$$

From Lemma 2.8, we know that

$$LJ = JL = 0.$$

So, we get

$$L^{+}L = 0 + \left[\begin{array}{c|c} Z - \frac{1}{n}ZJ - \frac{1}{n}JZ & -\frac{1}{n}ze \\ \hline -\frac{1}{n}e^{T}Z & O \end{array}\right]L$$

Also, we have

$$Z = L(\bar{n}\,)^{-1}, \quad e = 1 \text{ and } e^{T} = 1^{T}$$

$$L^{+}L = \left[\begin{array}{c|c} L(\bar{n}\,)^{-1} - \frac{1}{n}L(\bar{n}\,)^{-1}J - \frac{1}{n}JL(\bar{n}\,)^{-1} & -\frac{1}{n}L(\bar{n}\,)^{-1}1 \\ \hline -\frac{1}{n}1^{T}L(\bar{n}\,)^{-1} & O \end{array}\right]L$$

$$= \left[\begin{array}{c|c} L(\bar{n}\,)^{-1}L(\bar{n}) - \frac{1}{n}L(\bar{n}\,)^{-1}JL(\bar{n}) - \frac{1}{n}JL(\bar{n}\,)^{-1}L(\bar{n}) & -\frac{1}{n}L(\bar{n}\,)^{-1}1L(\bar{n}) \\ \hline -\frac{1}{n}1^{T}L(\bar{n}\,)^{-1}L(\bar{n}) & 1 - \frac{1}{n} \end{array}\right]$$

We know that

$$L(\bar{n}\,)^{-1}L(\bar{n}) = I.$$

So,

$$L^{+}L = \left[\begin{array}{c|c} I - \frac{1}{n}L(\bar{n}\,)^{-1}O - \frac{1}{n}JI & -\frac{1}{n}I1 \\ \hline -\frac{1}{n}1^{T}I & 1 - \frac{1}{n} \end{array}\right]$$

$$= \left[\begin{array}{c|c} I - \frac{1}{n}J & -\frac{1}{n}1 \\ \hline -\frac{1}{n}1^T & 1 - \frac{1}{n} \end{array}\right] = I - \frac{J}{n} = LL^+$$

Hence proved. □

The new formula for computing Kirchhoff Index is given in the following theorem.

**Theorem 2.12.** $Kf(G) = nTr(Z) - \sum_{k,l} z_{kl}$ (2.41)

*where Z is the inverse of the Laplacian matrix obtained by deleting any ith row and ith column, and* $\sum_{k,l} z_{kl}$ *is the sum of all the elements of matrix Z (note that Z =* $Y^{-1}$*).*

*Proof*. Using equation (2.40) we can calculate the (*i*, *j*)th entry of pseudoinverse $L^+$of the Laplacian matrix *L* in terms of the elements of the matrix *Z*:

$$l_{ij}^+ = \begin{cases} \frac{\sum_{k,l} z_{kl}}{n^2} + z_{ij} - \frac{1}{n}\sum_k z_{kj} - \frac{1}{n}\sum_l z_{il} & , \quad i \neq n, \ j \neq n \\ \frac{\sum_{k,l} z_{kl}}{n^2} - \frac{1}{n}\sum_k z_{kj} & , \quad i = n, \ j \neq n \\ \frac{\sum_{k,l} z_{kl}}{n^2} - \frac{1}{n}\sum_l z_{il} & , \quad i \neq n, \ j = n \\ \frac{\sum_{k,l} z_{kl}}{n^2} & , \quad i = n, \ j = n \end{cases} \qquad (2.42)$$

where,

$\sum_{k,l} z_{kl}$ is the sum of all the elements of the matrix *Z*

$\sum_k z_{kj}$ is the sum of the elements of the $k^{th}$ rows of the matrix *Z*

$\sum_l z_{il}$ is the sum of the elements of the $l^{th}$ columns of the matrix *Z*

Now using equation (2.37) and (2.40), we get

$$Kf(G) = nTr(L^+) = n\left(\sum_{i=1}^{n-1} l_{ii}^+ + l_{nn}^+\right). \qquad (2.43)$$

The trace of the pseudoinverse $L^+$ of the Laplacian matrix satisfies

$$Tr(L^{+}) = \sum_{i=1}^{n-1} l_{ii}^{+} + l_{nn}^{+}. \tag{2.44}$$

From (2.42) we get

$$l_{ii}^{+} = \frac{\sum_{k,l} z_{kl}}{n^2} + z_{ii} - \frac{2}{n}\sum_{l} z_{il} \tag{2.45}$$

$$l_{nn}^{+} = \frac{\sum_{k,l} z_{kl}}{n^2} \tag{2.46}$$

Now using (2.44), (2.45) and (2.46), we get

$$Tr(L^{+}) = \sum_{i=1}^{n-1}\left(\frac{\sum_{k,l} z_{kl}}{n^2} + z_{ii} - \frac{2}{n}\sum_{l} z_{il}\right) + \frac{\sum_{k,l} z_{kl}}{n^2}. \tag{2.47}$$

Note that $\sum_{i=1}^{n-1}(\sum_{l} z_{il}) = \sum_{k,l} z_{kl}$ (sum of all elements of matrix $Z$)

Thus,

$$Tr(L^{+}) = \frac{1}{n^2}(n-1)\sum_{k,l} z_{kl} + \sum_{i=1}^{n-1} z_{ii} - \frac{2}{n}\sum_{k,l} z_{kl} + \frac{\sum_{k,l} z_{kl}}{n^2}.$$

After simplification, we get

$$Tr(L^{+}) = \sum_{i=1}^{n-1} z_{ii} - \frac{\sum_{k,l} z_{kl}}{n}\,. \tag{2.48}$$

From (2.43) and (2.48), we get

$$Kf(G) = n\sum_{i=1}^{n-1} z_{ii} - \sum_{k,l} z_{kl}\,. \tag{2.49}$$

We know,

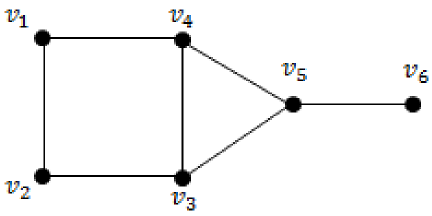
$$\sum_{i=1}^{n-1} z_{ii} = Tr(Z)\,. \tag{2.50}$$

The required result follows from (2.49) and (2.50) as

$$Kf(G) = nTr(Z) - \sum_{k,l} z_{kl}. \qquad \square$$

The following example demonstrates the calculation of the Kirchhoff Index by first using the Moore-Penrose Pseudoinverse and then by using our new formula.

**Example 2.13.** Figure 2.9 shows an unweighted graph $G$ with six nodes and its Laplacian matrix.



$$L = \begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 & 0 \\ -1 & 0 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

$(a)$ $(b)$

Figure 2.9: Laplacian matrix. (a) A graph G with six nodes. (b) Laplacian matrix L of graph G.

**Kirchhoff index using Moore-Penrose pseudo-inverse:**

First, we find the Moore-Penrose pseudoinverse of Laplacian matrix $L$ given in Figure 2.9($b$) by using formula (2.10):

$$L^+ = \left( \begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 & 0 \\ -1 & 0 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} + \frac{1}{6} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \right)^{-1}$$

$$- \frac{1}{6} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$L^{+} = \begin{bmatrix} 0.487 & 0.123 & -0.074 & 0.017 & -0.195 & -0.362 \\ 0.123 & 0.487 & 0.017 & -0.074 & -0.195 & -0.362 \\ -0.074 & 0.017 & 0.275 & 0.002 & -0.028 & -0.195 \\ 0.017 & -0.074 & 0.002 & 0.275 & -0.028 & -0.195 \\ -0.195 & -0.195 & -0.028 & -0.028 & 0.305 & 0.138 \\ -0.362 & -0.362 & -0.195 & -0.195 & 0.138 & 0.972 \end{bmatrix}$$

The trace of Moore-Penrose pseudoinverse is

$$Tr(L^{+}) = \sum_{i=1}^{n} l_{ii} = 2.801.$$

Let $Kf(G)$ be the Kirchhoff index of the graph given in Figure. 2.9(a). Now using (2.35) we can calculate Kirchhoff index $Kf(G)$ as

$$Kf(G) = 6 * 2.801 = 16.8.$$

Next, we calculate Kirchhoff Index $Kf(G)$ by using $Z$ $(\text{i.e.}, Z = L(\bar{n})^{-1})$.

**Kirchhoff index using our new formula:**

The matrix $Z$ of graph $G$ for Laplacian matrix $L$ in Figure 2.8 is

$$Z = \begin{bmatrix} 2.182 & 1.818 & 1.455 & 1.545 & 1 \\ 1.818 & 2.182 & 1.545 & 1.455 & 1 \\ 1.455 & 1.545 & 1.636 & 1.364 & 1 \\ 1.545 & 1.455 & 1.364 & 1.636 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

In order to find the Kirchhoff index $Kf(G)$, we calculate the trace of matrix $Z$ and the sum of all the elements of matrix $Z$:

$$Tr(Z) = 8.63$$

$$\sum_{k,l} z_{kl} = 35$$

Using (2.41), the Kirchhoff Index $Kf(G)$ is

$$Kf(G) = 6 * 8.63 - 35 = 16.8.$$

In the next section, we establish the formula in Theorem 2.1 using standard electrical circuit theoretic arguments based on the properties of the *n*-port resistance networks.

## 2.5 Kirchhoff Index using Topological Formulas for Network Functions

The formula in Theorem for Kirchhoff Index shows that not all the effective resistances are independent. That is, one can obtain Kirchhoff Index using only the matrix *Z*, whose diagonal entries are a subset of $(n-1)$ effective resistances. The off-diagonal entries in *Z* relate these $n-1$ effective resistances to the remaining effective resistances.

Consider a graph *G* of the network obtained by replacing each edge in the network by a resistance of one ohm. Let $Y = [y_{ij}]$ denote the node admittance matrix of *G* with node *n* as the reference or datum node.

Note that *Y* is a square matrix of order $n-1$ and it is the matrix obtained by removing the $n^{th}$ row and the $n^{th}$ column from the Laplacian matrix of $L$.

Note that $Z = Y^{-1}$.

As we have seen before,

$$r_{ij} = \frac{\tau_{i,j}}{\tau(G)}$$

However,

$$\begin{aligned} \tau_{ij} &= \tau_{i,nj} + \tau_{in,j} \\ &= \{\tau_{i,n} - \tau_{ij,n}\} + \{\tau_{j,n} - \tau_{ij,n}\} \\ &= \tau_{i,n} + \tau_{j,n} - 2\tau_{ij,n} \end{aligned}$$

Dividing by $\tau(G)$ both sides of the above equation we get

$$\frac{\tau_{ij}}{\tau(G)} = \frac{\tau_{i,n}}{\tau(G)} + \frac{\tau_{j,n}}{\tau(G)} - \frac{2\tau_{ij,n}}{\tau(G)}$$

$$r_{ij} = r_{i,n} + r_{j,n} - 2z_{ij} \tag{2.51}$$

Since each $r_{j,n}$ appears $n-1$ times on the right-hand side of the sum $\sum_{i,k>i} r_{i,k}$, we get

$$\sum_{i,k>i} r_{i,k} = (n-1)\sum_{j=1}^{n-1} r_{j,n} - 2\sum_{i,k>i} z_{ik}$$

$$= (n-1)\textstyle\sum_{j=1}^{n-1} r_{j,n} + \sum_{j=1}^{n-1} r_{j,n} - \left(\sum_{j=1}^{n-1} r_{j,n} + 2\sum_{i,k>i} z_{ik}\right)$$

$$Kf(G) = n\sum_{j=1}^{n-1} r_{j,n} - \left(\sum_{j=1}^{n-1} r_{j,n} + 2\sum_{i,k>i} z_{ik}\right)$$

The above is the same as

$$Kf(G) = n\sum_{i=1}^{n-1} z_{ii} - \left(\sum_{i,l} z_{il}\right)$$

## 2.6 Kirchhoff Index using Circuit Theoretic Concepts

We now give another proof of equation (2.51) using circuit-theoretic principles.

Consider again the description of an $(n+1)$- node network as given by equation

$$YV = I$$

when node 0 is chosen as the reference node (see equation 2.20). If we are interested in the description of the network as viewed across the ports (1, 0) , (2, 0) then it is equivalent to setting $I_2 = \cdots = I_n = 0$ . See Figure 2.10.

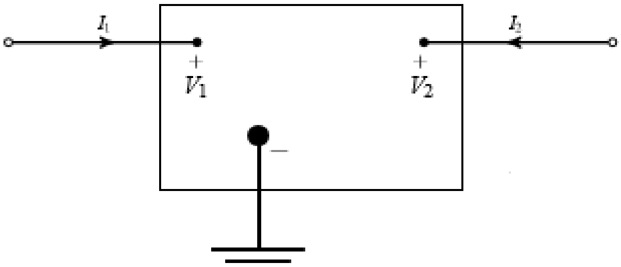


Figure 2.10: A 2-port network.

We then get

$$\begin{bmatrix} I_1 \\ I_2 \\ \hline 0 \\ \vdots \\ 0 \end{bmatrix} = \left[\begin{array}{c|c} Y_{11} & Y_{12} \\ \hline Y_{21} & Y_{22} \end{array}\right] \begin{bmatrix} V_1 \\ V_2 \\ \hline V_3 \\ \vdots \\ V_n \end{bmatrix}$$

Solving the above

$$\begin{pmatrix} I_1 \\ I_2 \end{pmatrix} = (Y_{11} - Y_{12}Y_{22}^{-1}Y_{21}) \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}$$

The matrix $Y_{11} - Y_{12}Y_{22}^{-1}Y_{21}$ is called a Schur Complements of *Y*. It is in fact the Laplacian matrix of the 3-node network obtained by repeated star-delta transformations at the nodes $2, \cdots, n$. See Figure 2.11, where *x, y, z* are the resistance of the equivalent network containing only nodes 1, 2, and 0.

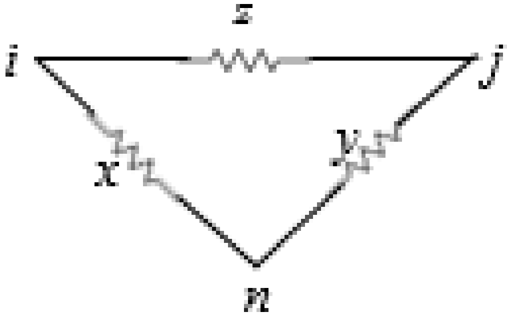


Figure 2.11: Three-node network.

Let the resistance distance between nodes *i* and *j* is denoted by $r_{ij}$. By using principles of circuit theory, we have

$$r_{in} = \frac{x(y+z)}{x+y+z} \tag{2.52}$$

$$r_{jn} = \frac{y(z+x)}{x+y+z} \tag{2.53}$$

$$r_{ij} = \frac{z(x+y)}{x+y+z} \tag{2.54}$$

The voltage across edge ( *j*, *n* ) when a unit current source is connected between *i* and *n* is denoted by $z_{ij}$.

$$z_{ij} = \frac{yx}{x+y+z} \tag{2.55}$$

Using (2.52), (2.53), (2.54) and (2.55), we get

$$\begin{aligned} r_{in} + r_{jn} - 2z_{ij} &= \frac{x(y+z)}{x+y+z} + \frac{y(z+x)}{x+y+z} - 2\frac{yx}{x+y+z} \\ &= \frac{xy + xz + yz + yz - 2yx}{x+y+z} \\ &= \frac{z(x+y)}{x+y+z} \\ &= r_{ij} \quad \text{(by equation 2.54)} \end{aligned}$$

## 2.7 Summary

In this chapter, we have given an overview of electrical networks along with the topological formulas for network functions. We also discussed the matrices of graph and their properties. Along with the Laplacian spectral graph theory we showed some known formulae of the Kirchhoff index using the Moore-Penrose pseudoinverse of the Laplacian matrix of a graph. We presented an interesting new formula for calculating the Kirchhoff index in terms of the matrix $Z$, to avoid the computational complexities and extraneous efforts of Moore-Penrose pseudoinverse. The matrix $Z$ is the inverse of the reduced Laplacian matrix $L(\bar{\imath})$.

Generalization of the Laplacian matrices and its relationship to the Kirchhoff index will be studied in the next chapter.

# Chapter 3

# Cutset Laplacian Matrix of a Graph and Kirchhoff Index

In chapter 2 we studied the relationship between the Laplacian matrix and the Kirchhoff index of a graph. Noting that the Laplacian matrix is defined by the reduced incidence matrix and the reduced incidence matrix is a submatrix of the cut matrix, in this chapter we generalize the notion of Laplacian matrix using the fundamental cutset matrix. We then develop two approaches to compute the Kirchhoff index. The first approach is based on a matrix transformation. To develop the second method, we define the concept of Kirchhoff polynomial of a graph which expresses Kirchhoff index using the elements of the resistance matrix. Since our discussion will be based on the fundamental cutset and fundamental circuit matrices, we begin with an introductory treatment of these concepts and their relationship with Kirchhoff voltage and current laws.

## 3.1 Cutsets

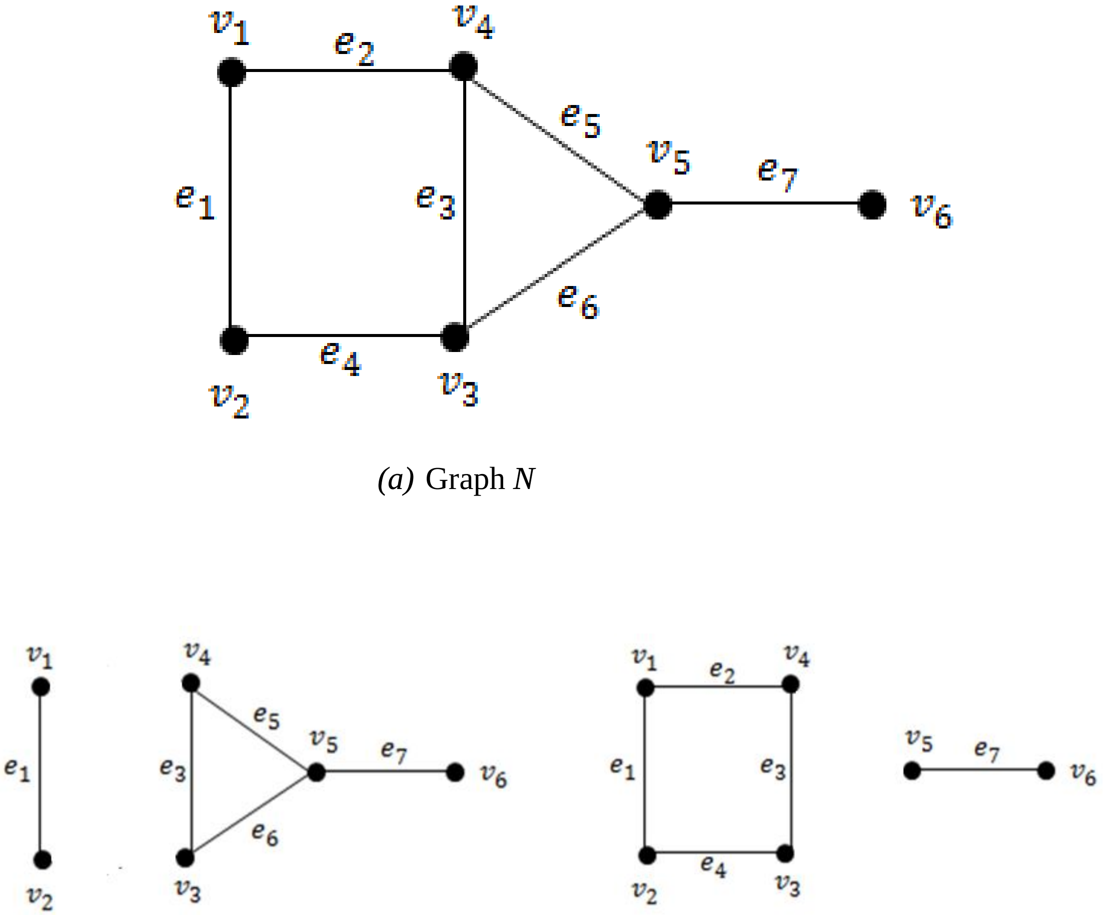
A graph *N* is said to be connected if there exists a path between every pair of vertices in *N*. For example, the graph of Figure 3.1 (a) is connected.

***Definition 3.1*** (Thulasiraman & Swamy, 1992)**.** A *cutset S* of a connected graph *N* is a minimal set of edges of *N* such that its removal from *N* disconnects *N*, that is, the graph *N* – *S* is disconnected.

For example, consider the subset $S_1 = \{e_2, e_4\}$ of edges of the graph *N* in Figure 3.1($a$). The removal of $S_1$ from graph *N* results in the graph $N_1 = N - S_1$ of Figure

3.1($b$). Graph $N_1$ is disconnected. Furthermore, the removal of any proper subset of $S_1$ cannot disconnect $N$. Thus $S_1$ is a cutset of $N$.

Consider next the subset $S_2 = \{e_5, e_6\}$. The graph $N_2 = N - S_2$ is shown in Figure 3.1($c$).



($a$) Graph $N$

($b$)$N_1$

($c$) $N_2$

Figure 3.1: Illustration of the definition of a cutset. (a) Graph N.
($b$) $N_1 = N - S_1$ , $S_1 = \{e_2, e_4\}$. ($c$) $N_2 = N - S_2$ , $S_2 = \{e_5, e_6\}$

## 3.2 Cuts

We now define the concept of a cut, which is closely to that of a cutset.

***Definition 3.2*** (Thulasiraman & Swamy, 1992)**.** Consider a connected graph *N* with vertex set *V*. Let $V_1$ and $V_2$be two mutually disjoint subsets of *V* such that $V = V_1 \cup V_2$; that is, $V_1$ and $V_2$ have no common vertices and together contain all the vertices of *V*. Then the set *S* of all those edges of graph *N* having one end vertex in $V_1$ and the other in $V_2$is called a *cut* of *N*. This is usually denoted by $\langle V_1, V_2 \rangle$.

Note that the cut $\langle V_1, V_2 \rangle$ of *N* is the minimal set of edges of *N* whose removal disconnects *N* into two graphs $N_1$and $N_2$, which are induces subgraphs of *N* on the vertex sets $V_1$ and $V_2$. $N_1$ and $N_2$may not be connected. If both these graphs are connected, then $\langle V_1, V_2 \rangle$ is also the minimal set of edges disconnecting *N* into exactly two components. Then by definition 3.1, $\langle V_1, V_2 \rangle$ is a cutset of *N*.

Suppose that for a cutset *S* of *N*, $V_1$ and $V_2$are, respectively, the vertex sets of the two components $N_1$and $N_2$of *N* – *S*. Then *S* is the cut $\langle V_1, V_2 \rangle$.

Thus, we have the following theorem.

**Theorem 3.3.**

1. *A cut* $\langle V_1, V_2 \rangle$ *of a connected graph N is a cutset of N if the induced subgraphs of N on vertex sets* $V_1$ *and* $V_2$ are connected.
2. *If S is a cutset of a connected graph N, and* $V_1$ *and* $V_2$*are the vertex sets of the two components of N –S, then S =* $\langle V_1, V_2 \rangle$.

Any cut $\langle V_1, V_2\rangle$ in a connected graph *N* contains a cutset of *N*, since the removal of $\langle V_1, V_2\rangle$ from *N* disconnects *N*. In fact, we can prove that a cut in a graph *N* is the union of some edge- disjoint cutsets of *N*. Formally, we state this in the following theorem.

**Theorem 3.4** *A cut in a connected graph N is a cutset or union of edge-disjoint cutsets of N.*

## 3.3 Fundamental Cutsets

In this section, we will show, how spanning tree can be used o define a set of fundamental cutsets.

Consider a spanning tree $T$ of a connected graph *N*. Let $b$ be a branch of T (*Note: The edges of a spanning tree T are called the branches of T* ). Now, the removal of the branch $b$ disconnects $T$ into exactly two components $T_1$ and $T_2$. Note that $T_1$ and $T_2$ are trees of *N*. Let $V_1$ and $V_2$, respectively, denote the vertex sets of $T_1$ and $T_2$. $V_1$ and $V_2$ together contain all vertices of *N*.

Let $N_1$ and $N_2$be, respectively, the induced subgraphs of *N* on the vertex sets $V_1$ and $V_2$. It can be seen that $T_1$ and $T_2$ are, respectively, the spanning trees of $N_1$ and $N_2$. Hence, $N_1$ and $N_2$ are connected. This, in turn, proves (Theorem 3.3) that the cut $\langle V_1$, $V_2\rangle$ is a cutset of *N*. This cutset is known as the *fundamental cutset* of *N* with respect to the branch b of the spanning tree $T$ of *N*. The set of all the $n$ −1 fundamental cutsets with respect to the $n$ −1 branches of a spanning tree $T$ of a connected graph *N* is known as the fundamental set of cutsets of *N* with respect to the spanning tree $T$. The rank

$\rho(N)$ of a connected $N$ is defined to be equal to $n - 1$. If $N$ has $p$ components, then $\rho(N) = n - p$.

Note that the cutset $\langle V_1, V_2\rangle$ contains exactly one branch, namely, the branch $b$ of $T$. All the other edges of $\langle V_1, V_2\rangle$ are links of $T$. This follows from the fact that $\langle V_1, V_2\rangle$ does not contain any edge of $T_1$ and $T_2$. Further, branch $b$ is not present in any other fundamental cutset with respect to $T$.

A graph $N$ and a set of fundamental cutsets of $N$ are shown in Figure 3.2.

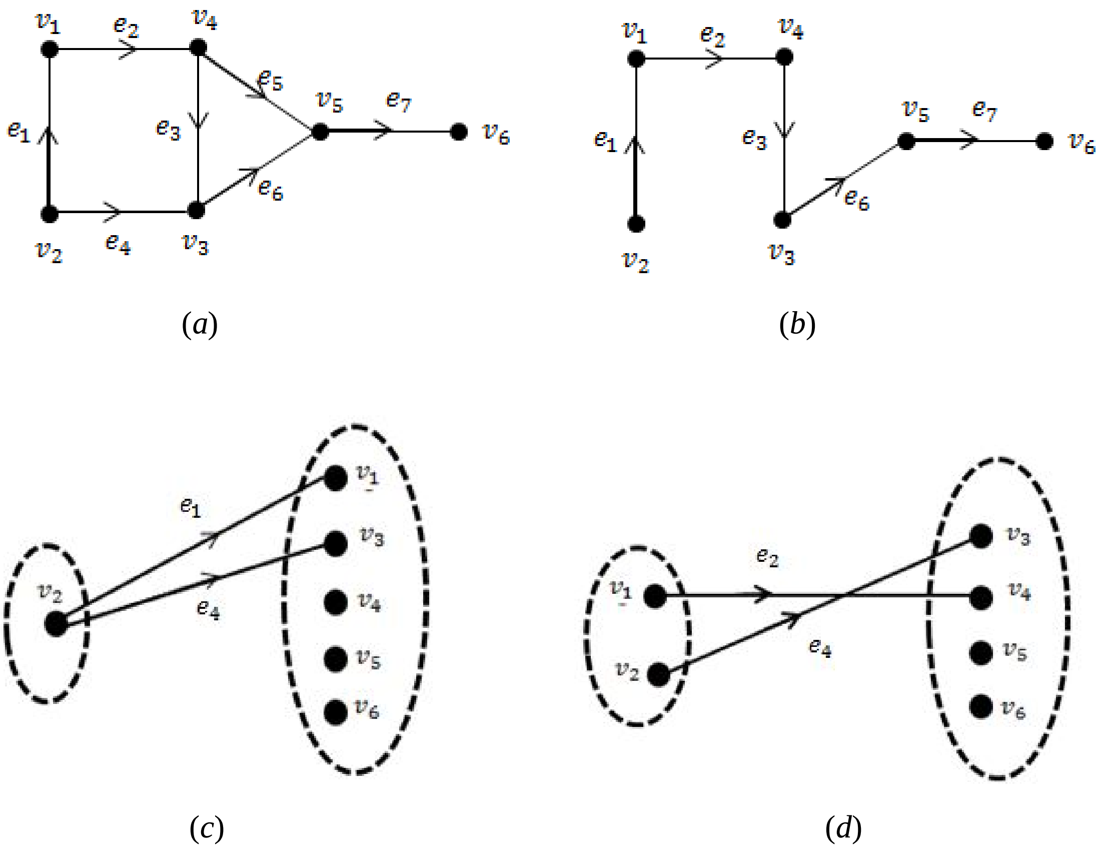
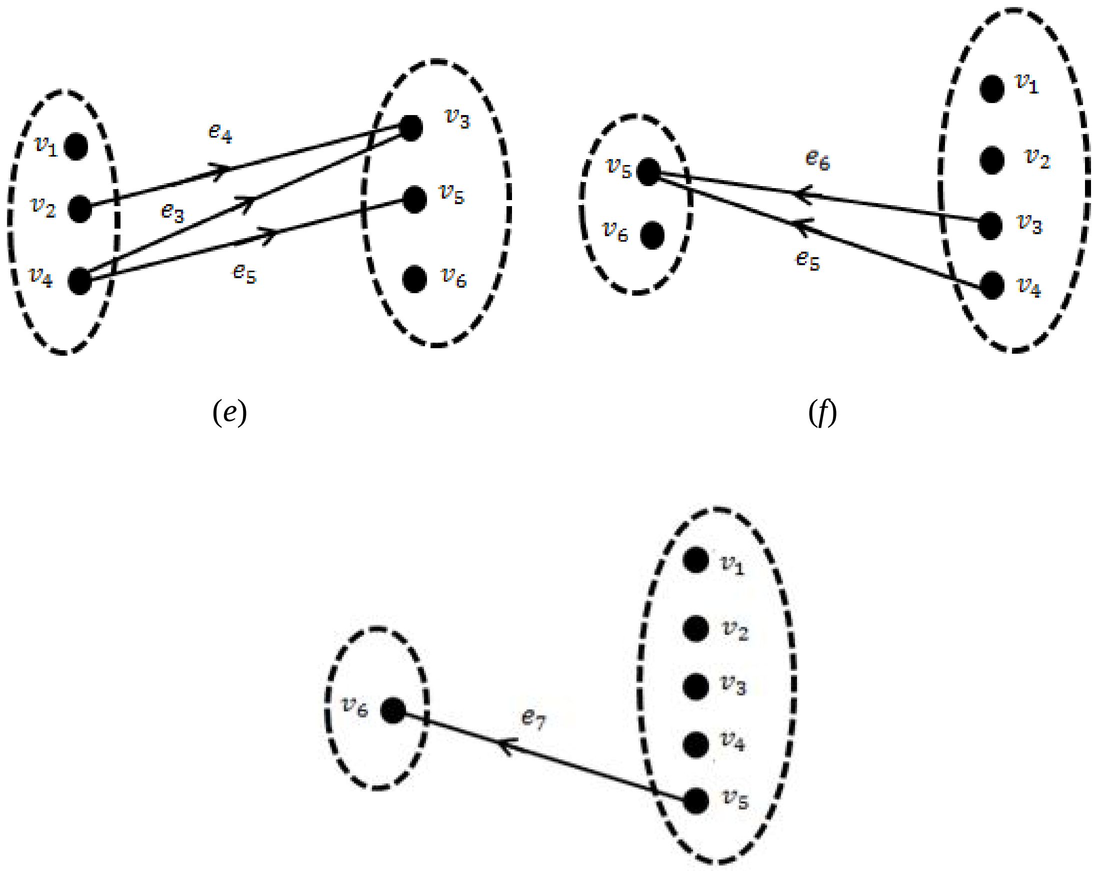


Figure 3.2: A set of fundamental cutsets of a graph. (*a*) Graph *N*. (*b*) Spanning tree *T* of *N*. (*c*) Fundamental cutset with respect to branch $e_1$. (*d*) Fundamental cutset with respect to branch $e_2$. (*e*) Fundamental cutset with respect to branch $e_3$ . (*f*) Fundamental cutset with respect to branch $e_6$ . (*g*) Fundamental cutset with respect to branch $e_7$.



($e$) ($f$)

($g$)

***Figure 3.2.*** (*Continued*)

It is obvious that removal of a cutset $S$ from a connected graph $N$ destroys all the spanning trees of $N$. A little thought will indicate that a cutset is a minimal set of edges whose removal from $N$ destroys all the spanning trees of $N$.

**Theorem 3.5.** *A cutset of a connected graph N contains at least one branch of every spanning tree of N.*

**Theorem 3.6.** *A set S of edges of a connected graph N is a cutset of N if and only if S is a minimal set of edges containing at least one branch of every spanning tree of N.*

## 3.4 Cut Matrix and Fundamental Cutset Matrix

To define the cut matrix of a directed graph we need to assign an orientation to each cut of the graph.

Consider a directed graph *N* = (*V*, *E*). If $V_a$ is a nonempty subset of *V*, then the set of edges connecting the vertices in $V_a$ to those in $\overline{V_a}$ is a cut, and this cut is denoted as $\langle V_a, \overline{V_a}\rangle$. The orientation of $\langle V_a, \overline{V_a}\rangle$ may be assumed to be either from $V_a$ to $\overline{V_a}$ or from $\overline{V_a}$ to $V_a$. Suppose we assume that the orientation is from $V_a$ to $\overline{V_a}$. Then the orientation of an edge in $\langle V_a, \overline{V_a}\rangle$ is said to agree with the orientation of the cut $\langle V_a, \overline{V_a}\rangle$ if the edge is oriented from a vertex in $V_a$ to a vertex in $\overline{V_a}$.

The *cut matrix* $Q_c$ = $[q_{ij}]$ of a graph *N* with *m* edges has *m* columns and as many rows as the number of cuts in *N*. The entry $q_{ij}$ is defined as follows:

*N* is *undirected*

$$q_{ij} = \begin{cases} 1, & \text{if the } j\text{th edge is in the } i\text{the cut ;} \\ 0, & \text{otherwise .} \end{cases}$$

*N* is *directed*

$$q_{ij} = \begin{cases} 1, & \text{if the } j\text{th edge is in the } i\text{the cut and its orientation agrees with} \\ & \text{the cut orientation ;} \\ -1, & \text{if the } j\text{th edge is in the } i\text{the cut and its orientation does not} \\ & \text{agrees with the cut orientation ;} \\ 0, & \text{otherwise .} \end{cases}$$

A row of $Q_c$ will be referred to as a *cut vector.*

Consider next any vertex *v*. The nonzero entries in the corresponding incidence vector represent the edges incident on *v*. These edges form the cut $\langle v,\ V - v\rangle$. If we assume that the orientation of this cut is from *v* to $V - v$, then we can see from the definitions of cut in section 3.2 and incidence matrices recall from chapter 2, that the row in $Q_c$ corresponding to the cut ⟨$v$, $V - v$⟩ is the same as the row in $A_c$ corresponding to the vertex v. Thus $A_c$ is a submatrix of $Q_c$.

**Theorem 3.7.** *Each row in the cut matrix $Q_c$ can be expressed, in two ways, as a linear combination the rows of the matrix $A_c$ . In each case, the nonzero coefficients in the linear combination are all* +1 *or all* $-1$.

**Theorem 3.8.** *The rank of the cut matrix $Q_c$ of an n-vertex connected graph N is equal to $n - 1$, the rank of N.*

As the above discussion and theorems show, the all –vertex incidence matrix $A_c$ is an important submatrix of the cut matrix $Q_c$. Next, we identify another important submatrix of $Q_c$, that is, *fundamental cutset matrix $Q_f$*.

We know from Section 3.3 that a spanning tree *T* of an *n*-vertex connected graph *N* defines a set of *n* − 1 fundamental cutsets—one fundamental cutset for each branch of *T*. The submatrix of $Q_c$ corresponding to these *n* −1 fundamental cutsets is known as the *fundamental cutset matrix $Q_f$* of *N* with respect to *T*.

Let $b_1, b_2, \dots, b_{n-1}$ denote the branches of tree *T*. Suppose we arrange the columns and the rows of $Q_f$ so that

1. For 1≤ *i* ≤ *n*−1, the *i*th column corresponds to the branch $b_i$.
2. The *i*th row corresponds to the fundamental cutset defined by $b_i$.

If, in addition, we assume that the orientation of a fundamental cutset is so chosen as to agree with that of the defining branch, then the matrix $Q_f$ can be displayed in a convenient form as follows:

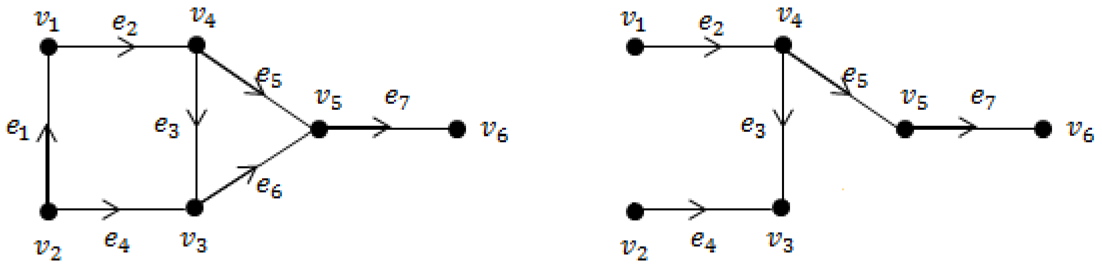
$$Q_f = [U|\, Q_{fc}] \tag{3.1}$$

where $U$ is the unit matrix of order $n-1$ and its columns correspond to the branches of $T$ and $Q_{fc}$ is the fundamental cutset of chords of $T$.

For example, the fundamental cutset matrix $Q_f$ of the connected graph of Figure 3.3($a$) with respect to the spanning tree $T = \{e_2, e_3, e_4, e_5, e_7\}$ is

$$Q_f = \begin{array}{c} \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} e_2 & e_3 & e_4 & e_5 & e_7 & e_1 & e_6 \end{array} \\ \left[\begin{array}{ccccc|cc} 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \end{array} \tag{3.2}$$

It is clear from (3.1) that the rank of fundamental cutset matrix $Q_f$ is equal to $n-1$, the rank of cut matrix $Q_c$. Thus, every cut vector (which may be a cutset vector) can be expressed as a linear combination of the fundamental cutset vectors.



($a$) ($b$)

Figure 3.3: ($a$) A directed Graph $N$. ($b$) Spanning tree $T$ of $N$.

## 3.5 Fundamental Circuit Matrix and Relationship with Fundamental Cutset Matrix

### 3.5.1 Fundamental circuits

Consider a spanning tree *T* of a connected graph *G*. Let the branches of *T* be denoted by $b_1, b_2, \cdots, b_{n-1}$ , and let the chords of *T* be denoted by $c_1, c_2, \cdots, c_{m-n+1}$, where $n$ is the number of vertices in *G* and *m* is the number of edges in *G*.

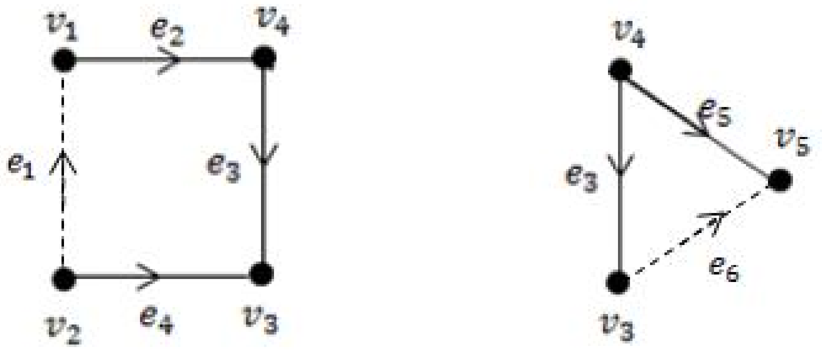
While *T* is acyclic, the graph $T \cup c_i$ contains exactly one circuit $C_i$. This circuit consists of the chord $c_i$ and those branches of *T* which lie in the unique path in *T* between the end vertices of $c_i$. The circuit $C_i$ is called the *fundamental circuit* of *G* with respect to the chord $c_i$ of the spanning tree *T*.

The set of all the $m - n + 1$ fundamental circuits $C_1, C_2, \cdots, C_{m-n+1}$ of *G* with respect to the chords of the spanning tree *T* of *G* is known as the *fundamental set of circuits* of *G* with respect to *T*. The nullity $\mu(G)$of a connected graph *G* is defined to be equal to $m - n + 1$. If *G* is not connected and has $p$ components, then $\mu(G) = m - n + p$.

An important feature of the fundamental circuit $C_i$ is that it contains exactly one chord, namely, chord $c_i$. Further, chord $c_i$ is not present in any other fundamental circuit with respect to *T*. For a given graph *G* and its spanning tree *T* in Figure 3.3, a set of fundamental circuits of G are shown in Figure 3.4.

### 3.5.2 Circuit matrix

A circuit can be traversed in one of two directions, clockwise or anticlockwise. The direction we choose for traversing a circuit defines its orientation.



$(a)$ Circuit $C_1$ $(b)$ Circuit $C_2$

Figure 3.4: Set of two fundamental circuits of $G$ (given in Figure 3.3($a$)) with respect to the spanning tree $T$ (given in Figure 3.3($b$)).

Consider an edge $e$ which has $v_i$ and $v_j$ as its end vertices. Suppose that this edge is oriented from $v_i$ and $v_j$ and that it is present in circuit $C$. Then we say that the orientation of $e$ agrees with the orientation of the circuit if $v_i$ appears before $v_j$ when we traverse $C$ in the direction specified by its orientation.

The *circuit matrix* $B_c = [b_{ij}]$ of a graph $G$ with $m$ edges has $m$ columns and as may rows as the number of circuits in $G$. The entry $b_{ij}$ is defined as follows:

*G is directed*:

$$b_{ij} = \begin{cases} 1, & \text{if the } j\text{th edge is in the } i\text{th circuit and its orientation agrees with the circuit orientation;} \\ -1, & \text{if the } j\text{th edge is in the } i\text{th circuit and its orientation does not agrees with the circuit orientation;} \\ 0, & \text{if the } j\text{th edge is not in the } i\text{th circuit.} \end{cases}$$

*G is undirected*:

$$b_{ij} = \begin{cases} 1, & \text{if the } j\text{th edge is in the } i\text{th circuit} \\ 0, & \text{otherwise} \end{cases}$$

A row of $B_c$ is called a circuit vector of $G$. Next, we identify an important submatrix of $B_c$.

### 3.5.3 Fundamental circuit matrix

Consider any spanning tree $T$ of a connected graph $G$ having $n$ vertices and $m$ edges. Let $c_1, c_2, \cdots, c_{m-n+1}$ be the chords of $T$. We know that these $m - n + 1$ chords define a set of $m-n+1$ fundamental circuits. The submatrix of $B_c$ corresponding to these fundamental circuits is known as the *fundamental circuit matrix* $B_f$ of $G$ with respect to the spanning tree $T$.

Suppose we arrange the columns and rows of $B_f$ so that

1. For $1 \leq i \leq m - n + 1$, the $i$th column corresponds to the chord $c_i$; and
2. The $i$th row corresponds to the fundamental circuit defined by $c_i$.

If, in addition, we choose the orientation of a fundamental circuit to agree with that of the defining chord, then the matrix $B_f$ can be written as

$$B_f = [U | \, B_{ft}] \tag{3.3}$$

where $U$ is the unit matrix of order $m - n + 1$ and its columns correspond to the chords of $T$.

For example, the fundamental circuit matrix of the graph of Figure 3.3 (a) with respect to the spanning tree $T = \{e_2, e_3, e_4, e_5, e_7\}$ is as given below:

$$B_f = \begin{array}{c|cc|ccccc|} & e_1 & e_6 & e_2 & e_3 & e_4 & e_5 & e_7 \\ e_1 & 1 & 0 & 1 & 1 & -1 & 0 & 0 \\ e_6 & 0 & 1 & 0 & 1 & 0 & -1 & 0 \end{array} \tag{3.4}$$

It is obvious from (3.3) that the rank of $B_f$ is equal to $m - n + 1$, the nullity $\mu(G)$ of $G$. Since $B_f$ is a submatrix of $B_c$, we get

$$\text{rank}(B_c) \geq m - n + 1$$

It is known (Thulasiraman & Swamy, 1992) that circuit and cutset vectors are orthogonal. That is,

$$Q_f B_f^t = 0. \tag{3.5}$$

Using this relation, we get

$$B_{ft} = -Q_{fc}^t \tag{3.6}$$

## 3.6 Kirchhoff's Laws and Fundamental Circuit and Cutset Matrices

Consider an electrical resistance network $G$. Let $T$ be a spanning tree of $G$. Then the fundamental cutset $Q_f$ matrix of $G$ has the form

$$\begin{array}{cc} \leftarrow \text{Branche} \rightarrow & \leftarrow \text{Chords} \rightarrow \\ Q_f = [ \quad U & | \quad Q_{fc} \quad ] \end{array}$$

and Kirchhoff's current law equations can be written as

$$Q_f I_e = 0 \tag{3.7}$$

that is

$$\begin{bmatrix} U & Q_{fc} \end{bmatrix} \begin{bmatrix} I_b \\ I_c \end{bmatrix} = 0 \tag{3.8}$$

where $I_b$ is the vector of branch currents and $I_c$ is the vector of chord currents. So

$$I_b = -Q_{fc} I_c \tag{3.9}$$

Similarly, we have

$$B_f = [\, B_{ft} \quad U \,] = [-\, Q_{fc}^t \quad U \,] \tag{3.10}$$

and Kirchhoff's voltage law equations can be written as

$$B_f V_e = 0 \tag{3.11}$$

that is

$$\begin{bmatrix} -Q_{fc}^t & U \end{bmatrix} \begin{bmatrix} V_b \\ I_c \end{bmatrix} = 0 \tag{3.12}$$

where $V_b$ is the vector of branch voltage and $V_c$ is the vector of chord voltage. So

$$V_c = -Q_{fc}^t V_b \tag{3.13}$$

## 3.7 Cutset Laplacian Matrix and Kirchhoff Index

Recall that the node-to-conductance matrix *Y*, also called the reduced Laplacian matrix, is given by

$$Y = AWA^t \tag{3.14}$$

where *A* is the reduced incidence matrix of *G* with respect to a specified reference vertex and *W* is the diagonal matrix of conductances of the elements of *G*.

Since each row of *A* represents a cut vector (set of edges incident on a node), we can generalize the notion of Laplacian matrix using fundamental cutset $Q_f$ in place of *A*.

***Definition 3.3. Generalized Laplacian matrix***

Let *T* be a spanning tree of a graph *G* and $Q_f$ be the fundamental cutset matrix of *G* with respect to *T*. If *W* is the diagonal matrix of edge conductances of *G*, then the *cutset Laplacian matrix* $Y_t$ of *G* is given by

$$Y_t = Q_f W Q_f^t \tag{3.15}$$

The matrix $Y_t$ is also called the *conductance matrix* of a multiport resistance network, as viewed from the branches of $T$ (called ports). The Matrix $Z_t = Y_t^{-1}$ is called the *resistance matrix* of the multiport network.

Each diagonal entry of $Z_t$ is the resistance $r_{ij}$ across the nodes $i$ and $j$ of the corresponding branch of the defining branch of $T$.

For example, the cutset Laplacian matrix $Y_t$ of the connected graph of Figure 3.3($a$) with respect to fundamental cutset matrix $Q_f$ given in (3.2) is

$$Y_t = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \end{bmatrix}$$

$$Y_t = \begin{bmatrix} 2 & 1 & -1 & 0 & 0 \\ 1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

So, we get,

$$Z_t = Y_t^{-1} = \left( \begin{bmatrix} 2 & 1 & -1 & 0 & 0 \\ 1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1}$$

$$= \begin{bmatrix} 8/11 & -2/11 & 3/11 & -1/11 & 0 \\ -2/11 & 6/11 & 2/11 & 3/11 & 0 \\ 3/11 & 2/11 & 8/11 & 1/11 & 0 \\ -1/11 & 3/11 & 1/11 & 7/11 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The (1, 1) entry of above matrix $Z_t$ is the resistance $r_{14}$. Also, element $Z_{ij} = V_j$, where $V_j$ is the voltage across the *j*th branch of *T* when a current source of unit value is connected across the nodes of the *i*th branch of *T*, as shown in Figure 3.5.

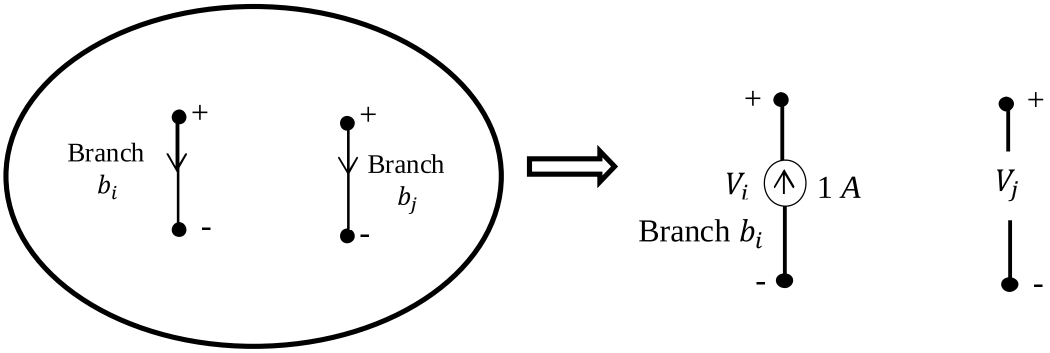


Figure 3.5: Voltage $\boldsymbol{V_j}$ across the *j*th branch when a current source of 1*A* is connected across the nodes of the *i*th branch.

## 3.8 Computing Kirchhoff Index: A Matrix Transformation Approach

In Chapter 2 we presented a formula to compute the Kirchhoff index using the elements of $(Y_n)^{-1}$, where $Y_n$ is the Laplacian matrix. In this section, we present a method to compute the Kirchhoff index from $Z_t$ using a matrix transformation approach.

Note that in view of our definition of the cutset Laplacian, $Y_n$ may be viewed as the cutset Laplacian with respect to the star stree $T_n$ (see Figure 3.6).

The matrix $(Y_t)^{-1} = (Q_f W Q_f^t)^{-1}$ specifies the relationship between the voltages across the branches of *T* and the currents injected through these branches (see Figure 3.7).

The matrix $Z_t = {Y_t}^{-1}$ relates $V_t$ and $I_t$ as

$$V_t = Z_t I_t \tag{3.16}$$

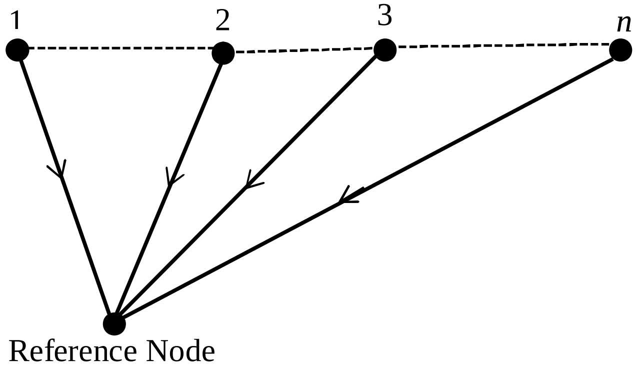


Figure 3.6: Star tree $\boldsymbol{T_n}$

If $Y_n$ is the Laplacian matrix when the star tree is used, then

$$V_n = Z_n I_n \tag{3.17}$$

where $Z_n = Y_n^{-1}$.

We can find the Kirchhoff index if $Z_n$ is known using (2.39).

Given $Z_t$, our interest is to determine $Z_n$ using a matrix transformation approach. We can then apply (2.38) on $Z_n$ to compute the Kirchhoff index.

Now we show how to relate $Z_n$ with $Z_t$.

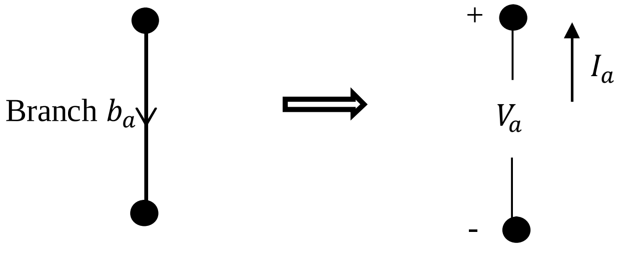


Figure 3.7: Voltages across the branches of $T$ and current injected through branches.

Note that each row of the reduced incidence matrix $A$ represents a cut. So the rows of $A$ represent $n-1$ linearly independent cutsets. This means each row of $Q_f$ can be written as a linear combination of the rows of $A$.

$$Q_f = [\ U \ | Q_{fc} \ ] = A_{11}^{-1}[\ A_{11} \quad A_{12}\ ]$$

where $A_{11}$ is the submatrix of columns of $A$ corresponding to the branches of $T$.

Now

$$\begin{aligned} Y_t &= Q_f G Q_f^t \\ &= A_{11}^{-1} A G (A_{11}^{-1} A)^t \\ &= A_{11}^{-1} (A G A^t)\ (A_{11}^{-1})^t \\ &= A_{11}^{-1} Y_n (A_{11}^{-1})^t \end{aligned}$$

So

$$\begin{aligned} Z_t &= Y_t^{-1} \\ &= A_{11}^t Y_n^{-1} A_{11} \\ &= A_{11}^t Z_n A_{11} \end{aligned}$$

So

$$Z_n = (A_{11}^{-1})^t Z_t (A_{11}^{-1}) \qquad (3.18)$$

Since

$$V_t = A_{11}^t V_n$$

$$V_n = (A_{11}^{-1})^t \qquad (3.19)$$

To compute $(A_{11}^{-1})^t$ we procced as follows. Consider any node $i$ and the corresponding node-to-datum voltage $V_i$.

Let $P_i$ be the path in $T$ from node $i$ to the datum node. Then $V_n$ is the algebraic sum of the voltages of the edges in $P_i$. For example, in Figure 3.8

$$v_1' = v_2 + v_3 - v_4\ .$$

Thus, we have the following formula to compute $(A_{11}^{-1})^t$

$$a_{ij}^{-1} = \begin{cases} 1, & \text{if the edge } e_j \in T \text{ lies in the path } P_i \text{ and is oriented in the direction} \\ & \text{from node } i \text{ to datum;} \\ -1, & \text{if the edge } e_j \in T \text{ lies in the path } P_i \text{ and is oriented in the direction} \\ & \text{from datum to node } i; \\ 0, & \text{if the edge } e_j \in T \text{ is not in path } P_i; \end{cases}$$

Consider the graph in Figure 3.3($a$) and the spanning tree $T$ in Figure 3.3($b$). The graph containing $T$ and the star tree $T_n$ (dashed lines) is shown in Figure 3.8.

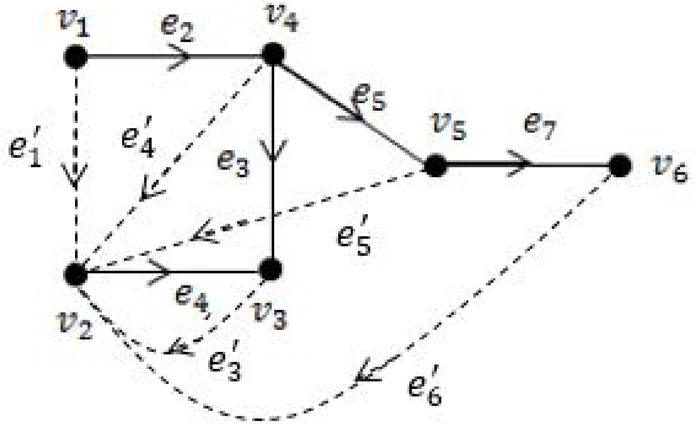


Figure 3.8: Graph containing spanning tree $T$ and star tree (dotted) given in Figure 3.3.

For the graph in Figure 3.8

$$\begin{bmatrix} v_1' \\ v_3' \\ v_4' \\ v_5' \\ v_6' \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 \\ 0 & 1 & -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix}$$

$$= (A_{11}^{-1})^t$$

One can easily see that $(A_{11}^{-1})^t$ is $-(B_{ft})$, a submatrix of $B_f$ of the graph in Figure 3.8.

Thus, we can rewrite (3.18) as

$$Z_n = B_{ft} Z_t B_{ft}^t$$

This is the transformation we have been looking for.

**Example 3.8.1.** For the graph in Figure 3.3($a$), the datum node is $v_2$. We get the following reduced Laplacian matrix by removing the 2nd row and 2nd column from the Laplacian Matrix of given graph.

$$Y_n = L(\bar{z}) = \begin{bmatrix} 2 & 0 & -1 & 0 & 0 \\ 0 & 3 & -1 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

**Calculating $Z_n$ by using reduced Laplacian matrix**

$$Z_n = (Y_n)^{-1} = \begin{bmatrix} 2 & 0 & -1 & 0 & 0 \\ 0 & 3 & -1 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}^{-1}$$

$$Z_n = \begin{bmatrix} 0.73 & 0.27 & 0.45 & 0.36 & 0.36 \\ 0.27 & 0.72 & 0.55 & 0.64 & 0.64 \\ 0.45 & 0.55 & 0.91 & 0.73 & 0.73 \\ 0.36 & 0.64 & 0.73 & 1.18 & 1.18 \\ 0.36 & 0.64 & 0.73 & 1.18 & 2.18 \end{bmatrix}$$

**Calculating $Z_n$ by using cutset Laplacian matrix**

$$Z_n = B_{ft} Z_t B_{ft}^t$$

$$Z_n = \begin{bmatrix} -1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 \\ 0 & -1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.72 & -0.18 & 0.27 & -0.09 & 0 \\ -0.18 & 0.55 & 0.18 & 0.27 & 0 \\ 0.27 & 0.18 & 0.72 & 0.09 & 0 \\ -0.09 & 0.27 & 0.09 & 0.64 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Z_n = \begin{bmatrix} 0.73 & 0.27 & 0.45 & 0.36 & 0.36 \\ 0.27 & 0.72 & 0.55 & 0.64 & 0.64 \\ 0.45 & 0.55 & 0.91 & 0.73 & 0.73 \\ 0.36 & 0.64 & 0.73 & 1.18 & 1.18 \\ 0.36 & 0.64 & 0.73 & 1.18 & 2.18 \end{bmatrix}$$

Using (2.41), Kirchhoff Index of $G$ is

$$KI(G) = 16.8.$$

## 3.9 Kirchhoff Polynomial of a Graph and a Formula for Kirchhoff Index

In this section, we determine a formula for the Kirchhoff index in terms of the elements of $Z_t$. We define a new concept called the *Kirchhoff polynomial* of a graph. This is a generalization of the formula in (2.39) in terms of the elements of $Z_n = (Y_n)^{-1}$, where $Y_n$ is the reduced Laplacian matrix of the graph.

***Definition* 3.4. Kirchhoff polynomial of a graph**.

Let $Y_t$ be the cutset Laplacian matrix of a resistance network *G* with respect to a spanning tree *T*. Let $Z_t = (Y_t)^{-1} = [z_{ij}]$. Kirchhoff polynomial of *G* is a polynomial $\sum_{i,j} C_{ij} z_{ij}$ that express Kirchhoff index of *G* in terms of the elements of $Z_t$. That is

$$\text{Kirchhoff index} = \sum C_{ij} z_{ij}.$$

We first determine a formula for each $r_{ij}$. Consider the path from vertex *i* to vertex *j* in the spanning tree *T*. To illustrate the ideas in our development, let this path be as given in Figure 3.9.

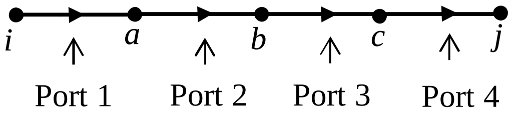


Figure 3.9: Path from vertex *i* to *j*.

For convenience, in Figure 3.9 the ports are oriented similarly. But in general, the ports can be oriented arbitrarily.

Consider now the 3-node equivalent representation of the graph as shown in Figure 3.10. This network can be obtained by repeated star-delta transformation at the remaining nodes.

Then by equation (2.51)

$$r_{bj} = r_{bc} + r_{jc} - 2V_{jc} = r_{bc} + r_{jc} + r_{bc} + 2z_{34}$$

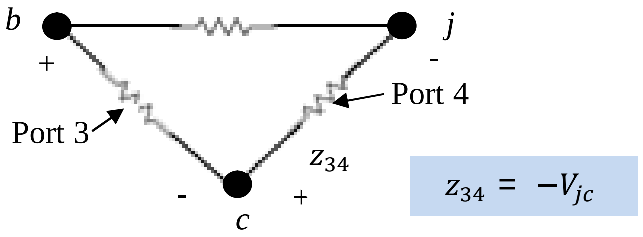


Figure 3.10: 3-node equivalent representation of the graph given in Figure 3.9.

Note that, if port 4 is oriented from $j$ to c, then

$$r_{bj} = r_{bc} + r_{cj} - 2z_{34}$$

as in equation (2.51).

Next consider $r_{aj}$, as shown in Figure 3.11,

$$r_{aj} = r_{ab} + r_{bj} - 2V_{bj}$$

$$= r_{ab} + r_{bj} + 2(z_{23} + z_{24})$$

In the above we have replaced $V_{bj}$ by $-z_{23} - z_{24}$ .

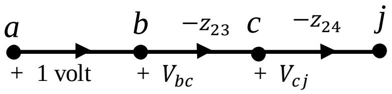


Figure 3.11

So

$$r_{aj} = r_{ab} + r_{bc} + r_{cj} + 2(z_{23} + z_{24}) + 2z_{34}$$

Continuing

$$r_{ij} = (r_{ia} + r_{ab} + r_{bc} + r_{cj}) + 2(z_{12} + z_{13} + z_{14}) + 2(z_{23} + z_{24}) + 2z_{34}.$$

Note that resistances $r_{ia}$, $r_{ab}$, $r_{bc}$ and $r_{cj}$ are diagonal elements of $Z_t$. For instance, $r_{ab}$ is the diagonal element $z_{22}$.

From the above we can see that the transfer resistance, say $z_{24}$ appears exactly once as $2z_{24}$ in the expressions of each of the resistance distances $r_{ij}$, $r_{aj}$ and $r_{bj}$. Generalizing this we can state that each $z_{kl}$ appears exactly once as $2z_{kl}$ in each $r_{xy}$ when the unique path in $T$ containing ports $x$ and $y$ spans ports $k$ and $l$ as shown in Figure 3.12. Each element $z_{ii}$ appears exactly once in each $r_{xy}$ when the unique path from $x$-to-$y$ in $T$ spans port $i$.

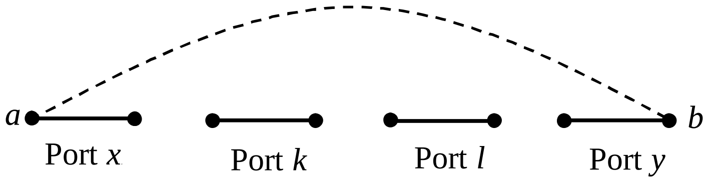


Figure 3.12

If $G_T$ is the complete graph on the vertices of $T$ then

$$C_{kl} = 2(\#\text{ number of edges in } G_T \text{ that span ports } k \text{ and } l\,), \quad \text{if } k \text{ and } l \text{ are similarly oriented}$$

$$= -2(\#\text{ number of edges in } G_T \text{ that span ports } k \text{ and } l\,), \quad \text{otherwise}$$

and

$$C_{kk} = (\#\text{ number of edges in the } f-\text{cutset defined by port } k)$$

Suppose we remove port $k$ from the tree then the $T$ will be disconnected into two trees. One of them will not contain port $l$. Let this tree be called $T_{kl}$. If we remove port $l$

from *T*, then the tree that does not contain port $k$ will be denoted by $T_{lk}$ . Then we can see that

$$C_{kl} = 2|T_{kl}| \cdot |T_{lk}|$$

Here $|T_{kl}|$ = # number of nodes in $T_{kl}$.

See Figure 3.13.

Also

$$C_{kk} = \left|T_k{}^{(1)}\right| \cdot \left|T_k{}^{(2)}\right|$$

where $T_k{}^{(1)}$ and $T_k{}^{(2)}$ are the two trees that result when port *k* is removed from the tree.

*Note*: $C_{kl} = C_{lk}$.

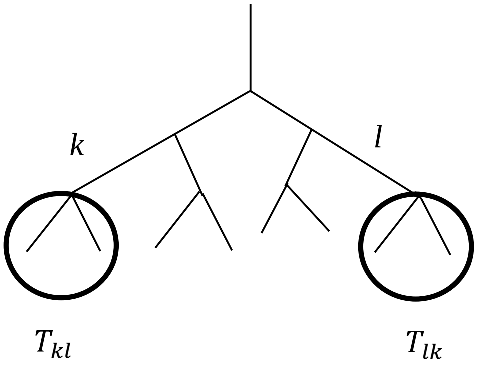


Figure 3.13

Summarizing the above discussion, we have the following theorem

**Theorem 3.9.** *Given a graph G with weight matrix W. Let T be a spanning tree of G. Let* $Z_t = [z_{ij}]$ *be the resistance matrix with respect to T. Then the Kirchhoff Index KI(G) is given by*

$$KI(G) = \sum_i C_{ii} Z_{ii} + \sum_{i,j} C_{ij} Z_{ij} \qquad (3.17)$$

$$= \sum_k \left|T_k{}^{(1)}\right| \cdot \left|T_k{}^{(2)}\right| + 2 \sum_{\substack{kl \\ k>l}} \pm |T_{kl}| \cdot |T_{lk}|.$$

In the case when $T$ is star tree

$T_i^{(1)} = 1$ for all $i$

$T_i^{(2)} = n-1$ for all $i$

$|T_{ij}| = 1$

$|T_{ji}| = 1$

Then

$C_{ii} = n-1$

$C_{ij} = \pm 1$, $i \neq j$, because all ports are dissimilarly oriented.

and

$$KI(G) = (n-1)Tr(Z_t) - 2 \sum_{i>j} z_{ij} \qquad (3.18)$$

This verifies formula (2.41) for the Kirchhoff index when the star tree is used in defining the cutset Laplacian matrix.

**Example 3.8.2.** For the graph given in Figure 3.8.

$$Z_t = \begin{bmatrix} 0.72 & -0.18 & 0.27 & -0.09 & 0 \\ -0.18 & 0.55 & 0.18 & 0.27 & 0 \\ 0.27 & 0.18 & 0.72 & 0.09 & 0 \\ -0.09 & 0.27 & 0.09 & 0.64 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The port numbers for Figure 3.8 are

Edge $e_2$ → Port 1, Edge $e_3$ → Port 2, Edge $e_4$ → Port 3,
Edge $e_5$ → Port 4, Edge $e_7$ → Port 5.

For the tree $T$ in Figure 3.3, $C_{ij}$ are

$C_{11} = 5$, $C_{12} = 2$, $C_{13} = -1$, $C_{14} = 2$, $C_{15} = 1$,
$C_{21} = 2$, $C_{22} = 8$, $C_{23} = -4$, $C_{24} = -4$, $C_{25} = -2$,
$C_{31} = -1$, $C_{32} = -4$, $C_{33} = 5$, $C_{34} = 2$, $C_{35} = 1$,
$C_{41} = 2$, $C_{42} = -4$, $C_{43} = 2$, $C_{44} = 8$, $C_{45} = 4$,
$C_{51} = 1$, $C_{52} = -2$, $C_{53} = 1$, $C_{54} = 4$, $C_{55} = 5$,

Using (3.17), we get Kirchhoff Index

$$KI = 16.8.$$

## 3.10 Summary

In this chapter, we have given an overview of the fundamental cutsets and fundamental circuits of a graph. We generalized the notion of Laplacian matrix using the fundamental cutset matrix. We presented two approaches to compute the Kirchhoff Index; first approach is based on a matrix transformation and the second approach used the concept of Kirchhoff polynomial of a graph.

In the next chapter, we generalize the notion of Kirchhoff index and study its relationship to Foster’s theorems.

# Chapter 4

# Weighted Kirchhoff Index of a Graph and Generalization of Foster's Theorems

In 1949, Foster (Foster, 1949) proved a theorem called Foster's First Theorem. This theorem gives an identity involving the sum of resistance distances. A graph-theoretic proof of this theorem was given in (Thulasiraman, et al., 1983) . In (Tetali, 1994) Tetali proved this theorem using certain results from the theory of Markov Chains. In 1961, Foster presented an extension of his first theorem (called Foster's second theorem). Building on Tetali's probabilistic approach, Palacios gave another proof of Foster's second theorem (Palacios, 2004). In this paper, Palacios also gave an extension of Foster's second theorem. In 2007, Cinkir (Cinkir, 2007) gave a generalization of all of Foster's theorems. These extensions are about the sum of resistance values over paths consisting of a certain number of edges. Connection between resistance distances and random walks on graph have been discussed in several works. See (Thulasiraman, et al., 2015) and (Doyle & Snell, 1984) for examples. See (Coppersmith, et al., 1990) and (Tetali, 1991) for the application of random walk and Foster's theorem in the design of on-line algorithms.

In this chapter, we provide further advances on the concept of Kirchhoff index. Our main contributions are the introduction of the concept of Weighted Kirchhoff index of a graph and its relationship to Foster's theorems. Specifically, we first show that Foster's theorems can be presented as results involving the sum of weighted $r_{ij}$'s $(w_{ij}r_{ij})$ when the weights are chosen appropriately. We then give a generalization of

Foster's theorems that retains the circuit-theoretic flavor and elegance of these theorems in section 4.3. We also present a dual form of Foster's first theorem in section 4.4.

## 4.1 Basic Concepts and Definitions

Consider a network $N$ of positive resistances. Let $V$ be the set of nodes in $N$. Let $n$ denote the number of nodes in $N$. We assume that the nodes are numbered 1, 2, …, $n$. So $V = \{1, 2, . . n\}$. Let $y_{ij}$ be the value of the conductance of the resistance element connecting nodes $i$ and $j$. Let $r_{ij}$ denote the input resistance of $N$ across the pair of nodes $i$ and $j$. $r_{ij}$ is also called the driving point resistance across nodes $i$ and $j$.

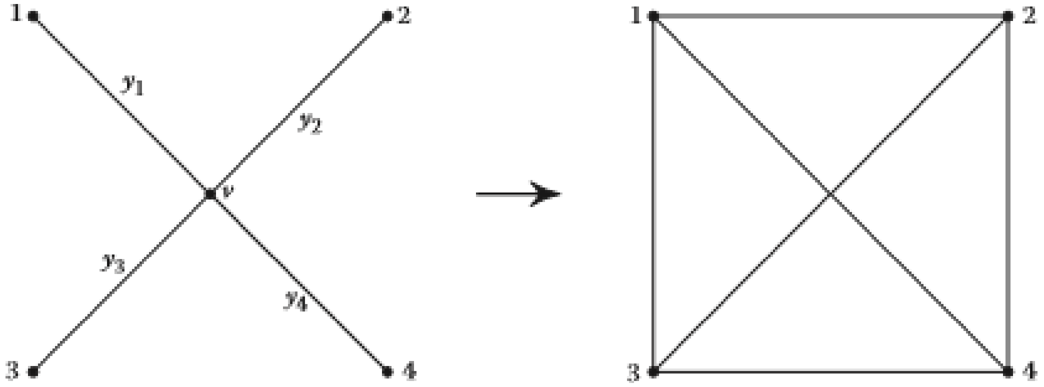
### 4.1.1 Star-Delta transformation

Consider a node $v$. Let $y_1, \dots, y_k$ be the conductances of the edges incident on $v$, with $1, 2, \dots, k$ denoting the other end nodes of these edges. Star-delta transformation at $v$ is the operation of removing node $v$ from $N$ and adding a new element $(i, j)$ with conductance $y_i y_j / d(v)$ for all $k \leq i, j \ \leq k$ (see Figure 4.1).

It is well known in circuit theory that the input resistance across nodes $i$ and $j$ in $N'$ is same as $r_{ij}$ in $N$ as long as these nodes remain in $N'$.

### 4.1.2 Multiple star-delta transformations

Let $D$ be a proper subset of nodes of $N$, that is, $D \subset V$. Suppose we perform star-delta transformations successively at the nodes in $D$, one at a time. Let $N(D)$ denote the resulting network. Clearly $N(D)$ has $n - k$ nodes when $k = |D|$. At the end of the multiple star-delta transformations, a new resistance element connecting $i$ and $j$ will be



$$y_{24} = \frac{y_2 y_4}{y_1 + y_2 + y_3 + y_4}$$

Figure 4.1: Star-delta transformation.

created in $N(D)$. Let the conductance value of the new element be $S_{ij}(D)$. Thus, the total value of the conductance of the elements connecting $i$ and $j$ in $N(D)$ will be $y_{ij} + S_{ij}(D)$. See Figure 4.2.

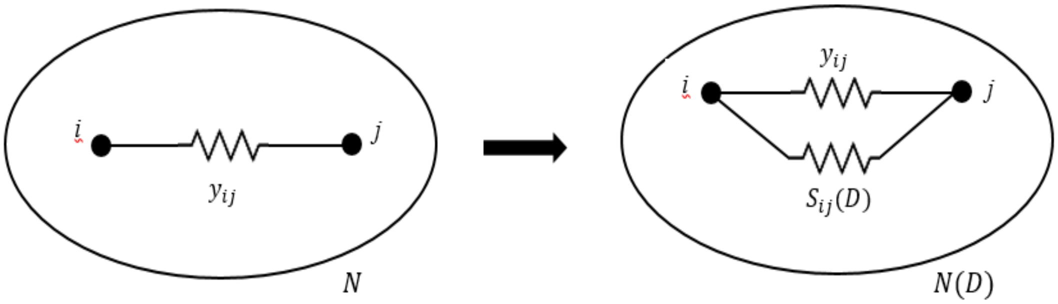


Figure 4.2: Multiple star-delta transformation.

Let

$$s_{ij}(k) = \sum_{\substack{D \subset V \\ |D|=k}} S_{ij}(D). \tag{4.1}$$

That is, $s_{ij}(k)$ is the sum of all $S_{ij}(D)$'s for all subsets of nodes of size $k$.

For example, consider a 5-node resistance network $N$ given in Figure 4.3. For this, there are ten 3-element subsets of nodes. These subsets are:

$\{a,b,c\}, \{a,b,d\}, \{a,b,e\}, \{a,c,d\}, \{a,c,e\}, \{a,d,e\}, \{b,c,d\}, \{b,c,e\}, \{b,d,e\}, \{c,d,e\}$

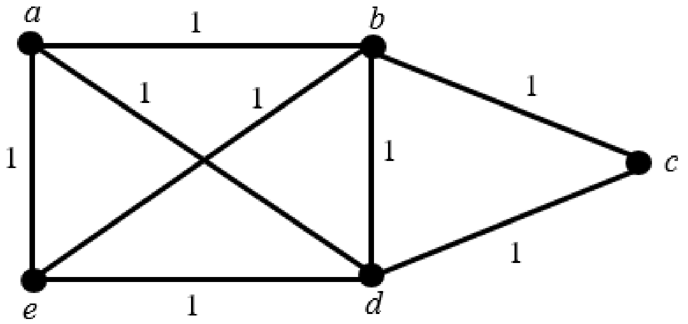


Figure 4.3: A 5-node resistance network $N$.

For each subset, $D$ of nodes, the corresponding network $N(D)$ is shown in Figure 4.4. In this figure, dotted edges indicate the new resistance elements along with the corresponding $S_{ij}(D)$'s. Then, using (4.1) we have

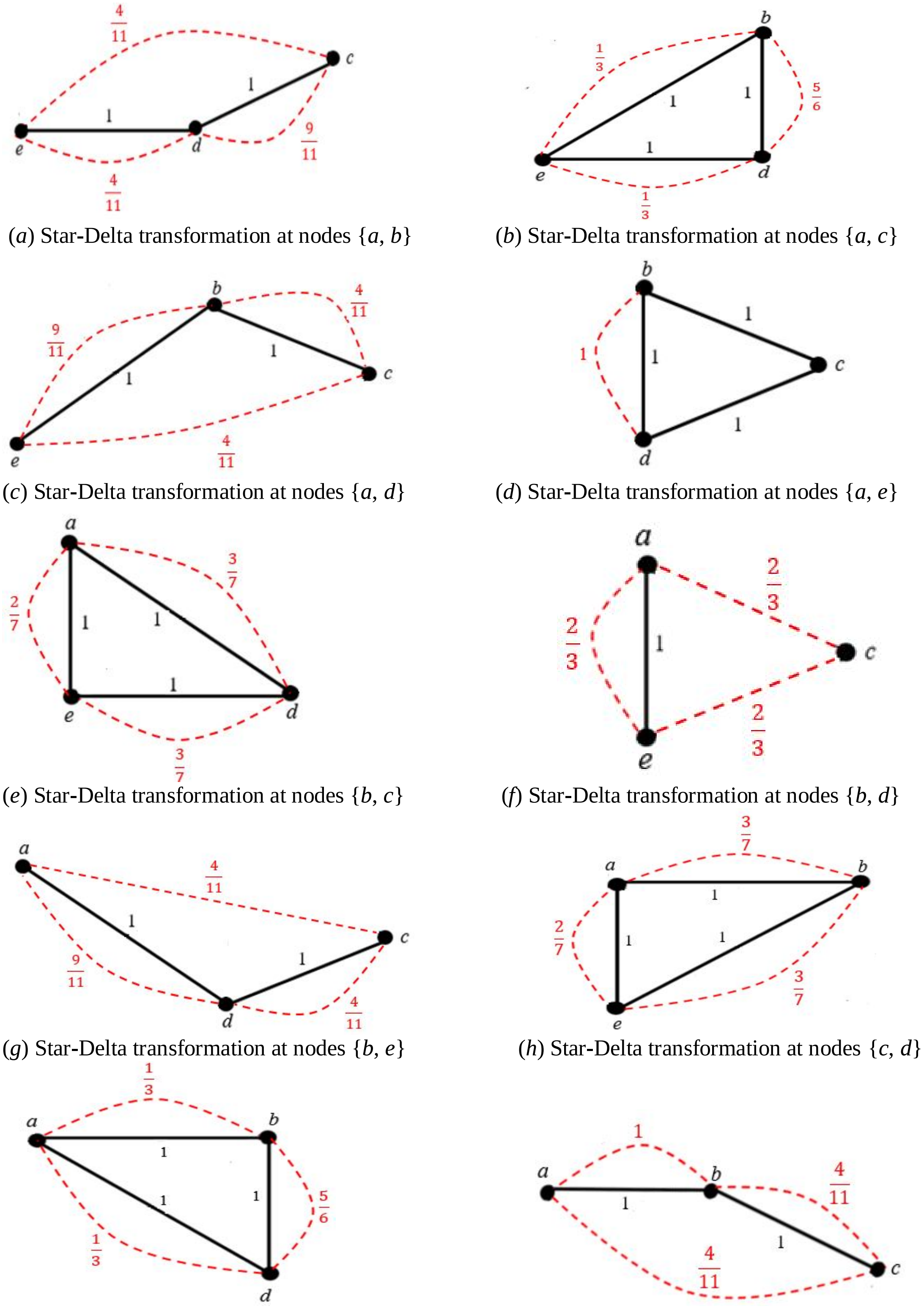
$$s_{ab}(2) = \frac{3}{7} + \frac{1}{3} + 1 = \frac{37}{21}, \qquad s_{ac}(2) = \frac{2}{3} + \frac{4}{11} + \frac{4}{11} = \frac{46}{33},$$

$$s_{ad}(2) = \frac{3}{7} + \frac{9}{11} + \frac{1}{3} = \frac{365}{231}, \qquad s_{ae}(2) = \frac{2}{7} + \frac{2}{3} + \frac{2}{7} = \frac{26}{21},$$

$$s_{bc}(2) = \frac{4}{11} + \frac{4}{11} = \frac{8}{11}, \qquad s_{bd}(2) = \frac{5}{6} + 1 + \frac{5}{6} = \frac{8}{3}$$

$$s_{be}(2) = \frac{1}{3} + \frac{9}{11} + \frac{3}{7} = \frac{365}{231}, \qquad s_{cd}(2) = \frac{4}{11} + \frac{4}{11} = \frac{8}{11},$$

$$s_{ce}(2) = \frac{4}{11} + \frac{4}{11} + \frac{2}{3} = \frac{46}{33}, \qquad s_{de}(2) = \frac{9}{11} + \frac{1}{3} + \frac{3}{7} = \frac{365}{231}$$



($a$) Star-Delta transformation at nodes $\{a, b\}$

($b$) Star-Delta transformation at nodes $\{a, c\}$

($c$) Star-Delta transformation at nodes $\{a, d\}$

($d$) Star-Delta transformation at nodes $\{a, e\}$

($e$) Star-Delta transformation at nodes $\{b, c\}$

($f$) Star-Delta transformation at nodes $\{b, d\}$

($g$) Star-Delta transformation at nodes $\{b, e\}$

($h$) Star-Delta transformation at nodes $\{c, d\}$

($i$) Star-Delta transformation at nodes $\{c, e\}$

($j$) Star-Delta transformation at nodes $\{d, e\}$

Figure 4.4: Corresponding network $N(D)$ for each subset $D$ of nodes.

## 4.2 Foster's Theorems

### 4.2.1 Foster's first theorem

Consider a resistance *N*. Let *N* have $n$ nodes and $m$ elements $e_1, e_2, \dots, e_m$. The resistance and conductance of each $e_i$ will be denoted by $z_i$ and $y_i \left(= \frac{1}{z_i}\right)$, respectively. Also, the two nodes of each $e_i$ will be denoted by $i_1$ and $i_2$. If $r_{i_1,i_2}$ denotes the effective resistance of *N* across the pair of nodes $i_1$ and $i_2$, then we have the following theorem due to Foster (Foster, 1949). For the sake of completeness, we provide a proof of this theorem repeated from (Thulasiraman, et al., 1983).

**Theorem 4.1 (Foster's First Theorem)**

$$\sum_{i=1}^{m} y_i r_{i_1,i_2} = n - 1 \tag{4.2}$$

*Proof.* Let *T* denote the set of all the spanning trees of *N* and, for each *i*, let $T_i$ denote the set of all the spanning 2-trees of *N* separating the nodes $i_1$ and $i_2$. That is, $T_i$ is the set of all the spanning trees of type $T_{i_1,i_2}$. Note that adding $e_i$ to a spanning 2-tree separating $i_1$ and $i_2$ will generate a spanning tree. Further, let $w(t)$ denote the conductance product of spanning tree *t* and $w(t_i)$ denote the conductance product of a spanning 2-tree $t_i$ separating $i_1$ and $i_2$. It is easy to see that if $t = t_i \cup e_i$ then

$$w(t) = y_i w(t_i).$$

If

$$W(T) = \sum_{t \in T} w(t)$$

and

$$W(T_i) = \sum_{t_i \in T_i} w(t_i)$$

then it is known (see 2.33, Chapter 2) that

$$r_{i_1,i_2} = \frac{W(T_i)}{W(T)}$$

Thus, to prove the theorem, we need to show that

$$\sum_{i=1}^{m} y_i W(T_i) = (n-1)W(T) \qquad (4.3)$$

or

$$\sum_{i=1}^{m} y_i \sum_{t_i \in T_i} w(t_i) = (n-1)\sum_{t \in T} w(t).$$

Consider any tree conductance product $w(t)$.We may assume, without loss of generality, that the spanning tree *t* contains the elements $e_1, e_2, \dots, e_{n-1}$. Then for every $i = 1, 2, \dots, n-1, t - e_i$ is a spanning 2-tree $t_i$ separating the nodes $i_1$ and $i_2$. So for every $i = 1, 2, \dots, n-1$,

$$w(t) = y_i w(t_i)$$

for some spanning 2-tree $t_i$. Thus, the conductance product $w(t)$ appears exactly once in each $y_i w(t_i)$, $i = 1, 2, \dots, n-1$. In other words, each $w(t)$ appears $n-1$ times in both sides of (4.2). The theorem follows since each $y_i w(t_i)$ corresponds to a unique $w(t)$. □

### 4.2.2 Foster’s second theorem

In this section, we state and prove Foster’s second theorem. This theorem is based on the operation of star-delta transformation which we define as follows.

Consider a node *v*. Let $y_1, \dots, y_k$ be the conductances of the edges incident on *v*, with $1, 2, \dots, k$ denoting the other end nodes of these edges. Recall that star-delta transformation at $v$ removes node $v$ from $N$ and adds a new element $(i, j)$ with conductance $y_i y_j / d(v)$ for all $k \leq i, j \ \leq k$ (see Figure 4.1).

Figure 4.5. illustrates an example to calculate the effective resistance $R_{12}$ between two vertices $v_1$ and $v_2$ by using Star-Delta transformation method.

The following theorem is by Foster (Foster, 1961).

**Theorem 4.2 (Foster's Second Theorem**) *Consider a resistance network N. For any pair of conductances $y_i$ and $y_j$ incident on common node v, let $r_{ij}$ denote the effective resistance across the two remaining nodes of $y_i$ and $y_j$. Let d(v) be the sum of the conductances of the elements incident on v. Then*

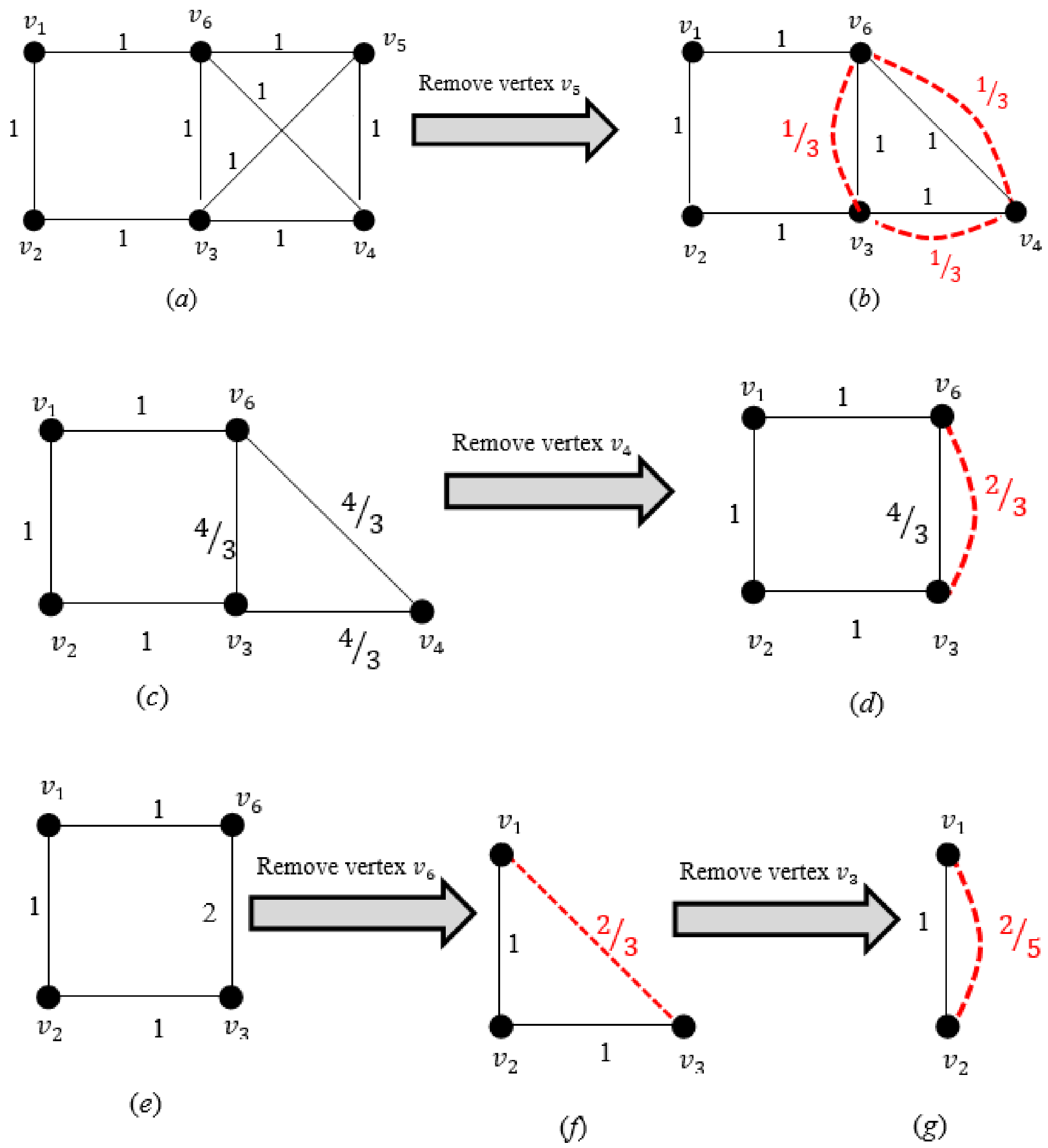
$$\sum_{v \in V} \sum_{i<j} r_{ij} y_{ij} = \sum_{i<j} \sum r_{ij} \frac{y_i y_j}{d(v)} = n - 2 \qquad (4.4)$$

*where the sum is extended over all pairs of adjacent elements incident on a common node v.*

*Proof.* Consider any node *v* in *N*. Star-delta transformation at *v* results in a network *N′* with $n - 1$ nodes. Applying Foster's First theorem to *N′* we get

$$\sum_{i<j \in N'} r_{ij} y_{ij} + \sum z_k y_k = n - 2. \qquad (4.5)$$

Here the first summation is over all pairs of elements of *N′* which reflect the new conductances created by star-delta transformation at node *v*. The second summation is over all conductances of *N* that are not connected to *v*. Note that $y_k$ is a conductance and $z_k$ is the effective resistance across the nodes of this conductance.



$$g_{12} = \frac{7}{5}, \text{ so } r_{12} = \frac{1}{g_{12}} = \frac{5}{7}$$

Figure 4.5: Calculating effective resistance distance between nodes $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$. Here, $g_{12}$ is the conductance between nodes $v_1$ and $v_2$.

Summing (4.5) over all the *n* vertices in *N*, we get

$$\sum_{i<j\in N'}\sum_{v} r_{ij}y_{ij} + \sum\sum z_k y_k = n(n-2).$$

The first sum is over all pairs of vertices adjacent to a common node *v* in N. The second sum is

$$\sum\sum z_k y_k = (n-2)\sum z_i y_i \tag{4.6}$$

because conductance $y_k$ appears exactly $n-2$ times in the double summation. So

$$\sum_{i<j\in N'}\sum_{v} r_{ij}y_{ij} = n(n-2)-(n-2)\sum z_i y_i$$

$$= n(n-2)-(n-2)(n-1) \text{ , applying Foster's First theorem}$$

$$= n-2$$

This completes the proof. □

## 4.3 Weighted Kirchhoff Index of a Resistance Network, Foster's Theorems, and Generalization

The Kirchhoff Index of a resistance network *N* is given by

$$KI(N) = \sum_{i<j} r_{ij}.$$

Suppose we associate a weight $w_{ij}$ to each $r_{ij}$. Then the corresponding weighted Kirchhoff index of *N* is defined as

$$WKI(N) = \sum_{i<j} w_{ij}r_{ij}.$$

Next, we present foster's two theorems stated in section 4.2 using the concept of weighted Kirchhoff index.

### 4.3.1 Foster's first theorem using weighted Kirchhoff index

**Theorem 4.3.** *If* $w_{ij} = y_{ij}$ *then*

$$WKI(N) = \sum_{i<j} y_{ij} r_{ij} = n - 1.$$

*Note*: $y_{ij} = 0$ if there is no resistance element connecting *i* and *j*. So, in that case, we get the original statement of Foster's theorem, namely,

$$\sum_{i \sim j} y_{ij} r_{ij} = n - 1.$$

*Note*: $i \sim j$ means there is an element connecting *i* and *j*.

### 4.3.2 Foster's second theorem using weighted Kirchhoff index

**Theorem 4.4.** *If* $w_{ij} = s_{ij}(1)$ *then*

$$WKI(N) = \sum_{i<j} s_{ij}(1) r_{ij} = n - 2.$$

We next state and prove the main contribution of this chapter that generalizes Foster's theorems.

### 4.3.3 Generalized Foster's theorem

**Theorem 4.5.** *If* $w_{ij} = s_{ij}(k),\ k \geq 1$ *then*

$$WKI(N) = \sum_{i<j} s_{ij}(k) r_{ij} = (n - k - 1)\binom{n-1}{k-1}$$

*Proof*. Consider a resistance network *N* of *n* nodes with nodes numbered 1, 2, …, *n*. Let $V = \{1, 2, \dots, n\}$. Let $D$ be a proper subset of $V$ and $|D| = k$. Then the network $N(D)$

that results after *Star-Delta Transformations* at the nodes of $D$ will have $n-k$ nodes. So, applying Foster's Theorem on $N(D)$, we get

$$\sum_{i<j}(y_{ij} + S_{ij}(D))r_{ij} = n-k-1. \tag{4.7}$$

Equation (4.7) can be rewritten as

$$\sum_{i<j} S_{ij}(D)r_{ij} + \sum_{i<j} y_{ij}r_{ij} = n-k-1. \tag{4.8}$$

Let us now write similar equations for all the $\binom{n}{k}$ subsets of $V$ of size $k$ and sum up both the right-hand side and left-hand side terms.

Then we get

$$\sum_{D\subset V}\sum_{i<j} S_{ij}(D)r_{ij} + \sum_{D\subset V}\sum_{i<j} y_{ij}r_{ij} = \binom{n}{k}(n-k-1). \tag{4.9}$$

Equation (4.9) can be rewritten as

$$\sum_{i<j} s_{ij}(k)r_{ij} + \sum_{D\subset V}\sum_{i<j} y_{ij}r_{ij} = \binom{n}{k}(n-k-1). \tag{4.10}$$

Consider the second term $\sum_{D\subset V}\sum_{i<j} y_{ij}r_{ij}$ in (4.9). In this summation, $y_{ij}r_{ij}$ will be present only if $D$ does not contain both $i$ and $j$. There are $\binom{n-2}{k}$ subsets of $V$ that satisfy this requirement. In all other cases, $y_{ij}r_{ij}$ will not be present. Thus, each term $y_{ij}r_{ij}$ appears exactly $\binom{n-2}{k}$ times in the second sum (4.9). So, we can rewrite (4.9) as

$$\sum_{i<j} s_{ij}(k)r_{ij} + \binom{n-2}{k}\sum_{i<j} y_{ij}r_{ij} = \binom{n}{k}\ (n-k-1).$$

That is

$$\textstyle\sum_{i<j} s_{ij}(k) r_{ij} + \binom{n-2}{k}(n-1) = \binom{n}{k}\ (n-k-1), \text{ by Theorem 4.3.}$$

So,

$$\begin{aligned}
\sum_{i<j} s_{ij}(k) r_{ij} &= (n-k-1)\binom{n}{k} - \binom{n-2}{k}(n-1) \\
&= (n-k-1)\left[\binom{n}{k} - \frac{(n-1)}{(n-k-1)}\binom{n-2}{k}\right] \\
&= (n-k-1)\left[\binom{n}{k} - \frac{(n-1)!}{k!\,(n-k-1)!}\right] \\
&= (n-k-1)\left[\binom{n}{k} - \binom{n-1}{k}\right] \\
&= (n-k-1)\left[\binom{n-1}{k-1}\right],
\end{aligned}$$

where the identity $\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$ is used. □

For example, the $WKI(N)$ of the 5-node resistance network *N* (Figure 4.3) for $k = 2$ is calculated below. Note that $|D| = k$. The resistance distance $r_{ij}$ for each pair of nodes for the 5-node network *N* (Figure 4.3) is

$r_{ab} = 0.475$, $r_{ac} = 0.875$, $r_{ad} = 0.475$, $r_{ae} = 0.500$,

$r_{bc} = 0.600$, $r_{bd} = 0.400$, $r_{be} = 0.475$, $r_{cd} = 0.600$,

$r_{ce} = 0.875$, $r_{de} = 0.475$.

By using the above calculated $r_{ij}$'s and $s_{ij}(2)$'s, we can calculate $r_{ij}s_{ij}(2)$ for each pair of nodes as given below:

$R_{ab}s_{ab}(2) = 0.837$, $R_{ac}s_{ac}(2) = 1.219$, $R_{ad}s_{ad}(2) = 0.750$, $R_{ae}s_{ae}(2) = 0.619$,

$R_{bc}s_{bc}(2) = 0.436$, $R_{bd}s_{bd}(2) = 1.066$, $R_{be}s_{be}(2) = 0.750$, $R_{cd}s_{cd}(2) = 0.436$,

$R_{ce}s_{ce}(2) = 1.219$, $R_{de}s_{de}(2) = 0.750$,

So,

$$WKI(N) = \sum_{i<j} s_{ij}(k)r_{ij} = 8.08 \cong 8$$

For *n* = 5 and *k* = 2, we have by Theorem 4.5:

$$\sum_{i<j} s_{ij}(k)r_{ij} = (n-k-1)\binom{n-1}{k-1} = 3\binom{4}{1} = 8,$$

verifying the result in Theorem 4.5.

## 4.4 Dual Form of Foster's First Theorem

Circuits and cutsets are dual concepts (Swamy & Thulasiraman, 1981). The cutset space (KCL equations) has dimension $n-1$, rank of the graph, and the circuit space (KVL equations) has dimension $m-n+1$, nullity of the graph. Here *m* is the number of resistance elements in *N*. Foster's theorem states that the weighted Kirchhoff index of a graph is $n-1$, the rank, when all weights are equal to unity. The question arises whether one could assign weights appropriately so that the corresponding weighted Kirchhoff index is equal to $m-n+1$, the nullity. We shall answer this question in the affirmative.

Note that the largest value that *k* can take in Theorem 4.5 is equal to *n* – 2, since at least two nodes are needed to define resistance distance.

**Theorem 4.6 (Dual of Foster's First Theorem).**

$$\sum_{\substack{i<j \\ i\sim j}} s_{ij}(n-2)r_{ij} = m-n+1 = \textit{nullity of graph } G$$

*Proof.* For $k = n-2$, we can get from Theorem 4.5 that

$$\sum_{i<j} s_{ij}(n-2)r_{ij} = \binom{n-1}{n-3}$$

$$= \binom{n-1}{2}$$

$$= \frac{(n-1)(n-2)}{2}$$

Rewriting the above, we get

$$\sum_{\substack{i<j \\ i\sim j}} s_{ij}(n-2)r_{ij} + \sum_{\substack{i<j \\ i\nsim j}} s_{ij}(n-2)r_{ij} = \frac{(n-1)(n-2)}{2}$$

where $i \sim j$ means that there is an edge connecting $i$ and $j$ .

Since $s_{ij}(n-2) = \frac{1}{r_{ij}}$ when $i \nsim j$, we get

$$\sum_{\substack{i<j \\ i\sim j}} s_{ij}(n-2)r_{ij} + m' = \frac{(n-1)(n-2)}{2}$$

where $m'$ is the number of resistance elements that are not in the network.

Since $m' = \frac{n(n-1)}{2} - m$ , we get

$$\sum_{\substack{i<j \\ i\sim j}} s_{ij}(n-2)r_{ij} = \frac{(n-1)(n-2)}{2} + m - \frac{n(n-1)}{2}$$

$$= m - n + 1$$

$$= \text{nullity of } G. \qquad \square$$

## 4.5 Summary

In this chapter, we first introduced the notion of Weighted Kirchhoff index of a graph. We then presented Foster's theorems in terms of the Weighted Kirchhoff index of a graph. Two specific choices of weights to be associated with resistance distance result in Foster's first and second theorems. A generalization of Fosters theorems was then discussed. Unlike the generalization in (Cinkir, 2011), our generalization retains the elegance and circuit-theoretic flavor of Foster's theorems. Our final result is to develop a dual form of Foster's first theorem. Since Foster's theorems capture the impact of path weights between nodes, we believe that our results provide a framework for the study of cascading failures using resistance distances.

# Chapter 5

# Computing Kirchhoff Index

## 5.1 Introduction

Kirchhoff Index is a structural descriptor of networks based on resistance distance. In this chapter, we discuss sequential and parallel algorithms for resistance distance by using Star-delta transformation. To study the properties of large networks, they are partitioned into clusters. The boundary nodes of the clusters connect them to other clusters in network. We propose a novel three-step approximation algorithm for Kirchhoff Index, by storing the resistance distance information of each cluster on its boundary nodes. The quality of the approximation algorithm depends on the density of the network.

Section 5.2 describes the graph partition using the metis software. Section 5.3 describes the Graphics Processing Units (GPU) and CUDA for parallel approach. In Section 5.4, we discuss the Star-Delta transformation algorithm using the series and parallel reduction. The sequential and parallel algorithms for finding the resistance distance are presented in Section 5.5. A novel approximation algorithm for resistance distance and Kirchhoff index is presented in Section 5.6.

## 5.2 Graph Partition using METIS

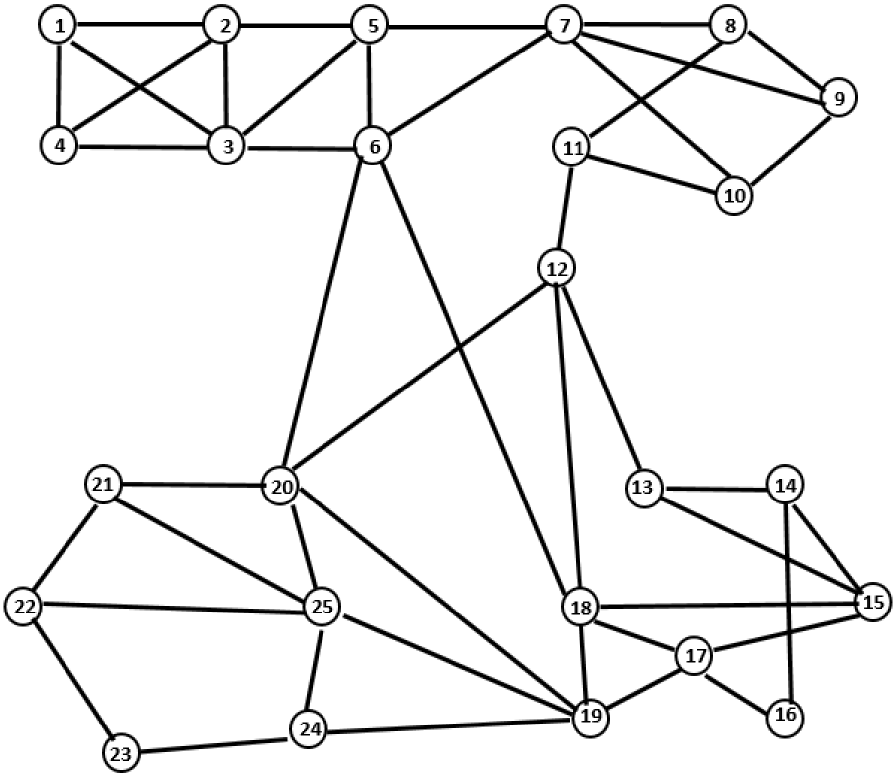
Metis (Karypis & Kumar, 2013) is a serial software package for partitioning large graphs. Metis consists of a fundamental library and a number of executable C programs. Metis software is freely distributed and has been developed at the Department of

Computer Science & Engineering at the University of Minnesota. Metis software can be downloaded directly from *http://www.cs.umn.edu*. The algorithms implemented in Metis are based on the multi-level graph paradigm (Karypis & Kumar, 2013). Metis uses KL algorithm developed by Kernighan-Lin (Kernighan & Lin, 1970) for graph partitioning.

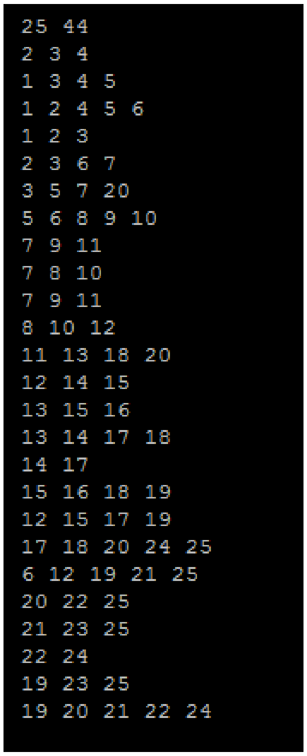
We used Metis 5.1.0 software for our experiments. For graph partitioning we used a stand-alone program, provided by Metis 5.x, called *gpmetis*. Gpmetis partitions a given graph into specified number of *clusters* or *parts*. The input graph is stored in a *graphfile* and the output of gpmetis is stored as *graphfile.part.nparts* where nparts is the number of parts or clusters the graph was partitioned into.

The input graph file and output file for an undirected graph $G$ are shown in Figure 5.1. The undirected graph $G$ given in Figure 5.1(a) consists of 25 nodes and 44 edges. The input graph file of graph $G$ with $n$ vertices and $m$ edges consists of $n + 1$ lines. The first line of input graph file is called *header line* and it contain the information about the number of nodes and number of edges of graph $G$. The remaining $n$ lines contain the information about the actual structure of the graph $G$. In particular, the $i$th line contains the information about the list of nodes, connected to node $i$.

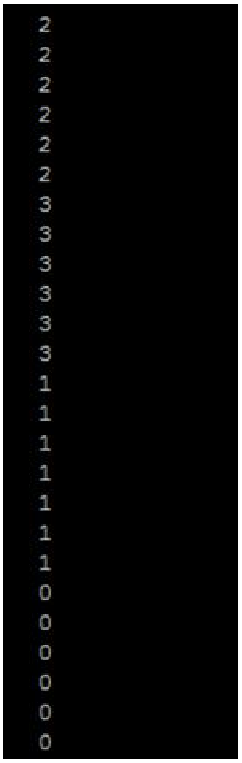
Figure 5.1($b$) illustrates the input graph file of graph $G$. The header line contains the information about size of graph as $n$ = 25 and $m$ = 44. The remaining lines represent all the nodes connected to a particular node. The output partition file of a graph $G$ consists of $n$ lines with a single number per line. The $i$th line in the output file represents the $i$th node of the graph and the number present at the $i$th line is the partition number where the $i$th node belongs to. Partitions are numbered from 0 to $k - 1$, where $k$ is the



(*a*)



```
25 44
2 3 4
1 3 4 5
1 2 4 5 6
1 2 3
2 3 6 7
3 5 7 20
5 6 8 9 10
7 9 11
7 8 10
7 9 11
8 10 12
11 13 18 20
12 14 15
13 15 16
13 14 17 18
14 17
15 16 18 19
12 15 17 19
17 18 20 24 25
6 12 19 21 25
20 22 25
21 23 25
22 24
19 23 25
19 20 21 22 24
```



```
2
2
2
2
2
2
3
3
3
3
3
3
1
1
1
1
1
1
1
0
0
0
0
0
0
```

(*b*) (*c*)

Figure 5.1: (*a*) Graph *G*. (*b*) Input graph file. (*c*) Output graph partition file.

the number of partition. Figure 5.1(*c*) shows the partition output file of graph *G*. The four clusters of the partition graph are shown in Figure 5.2.

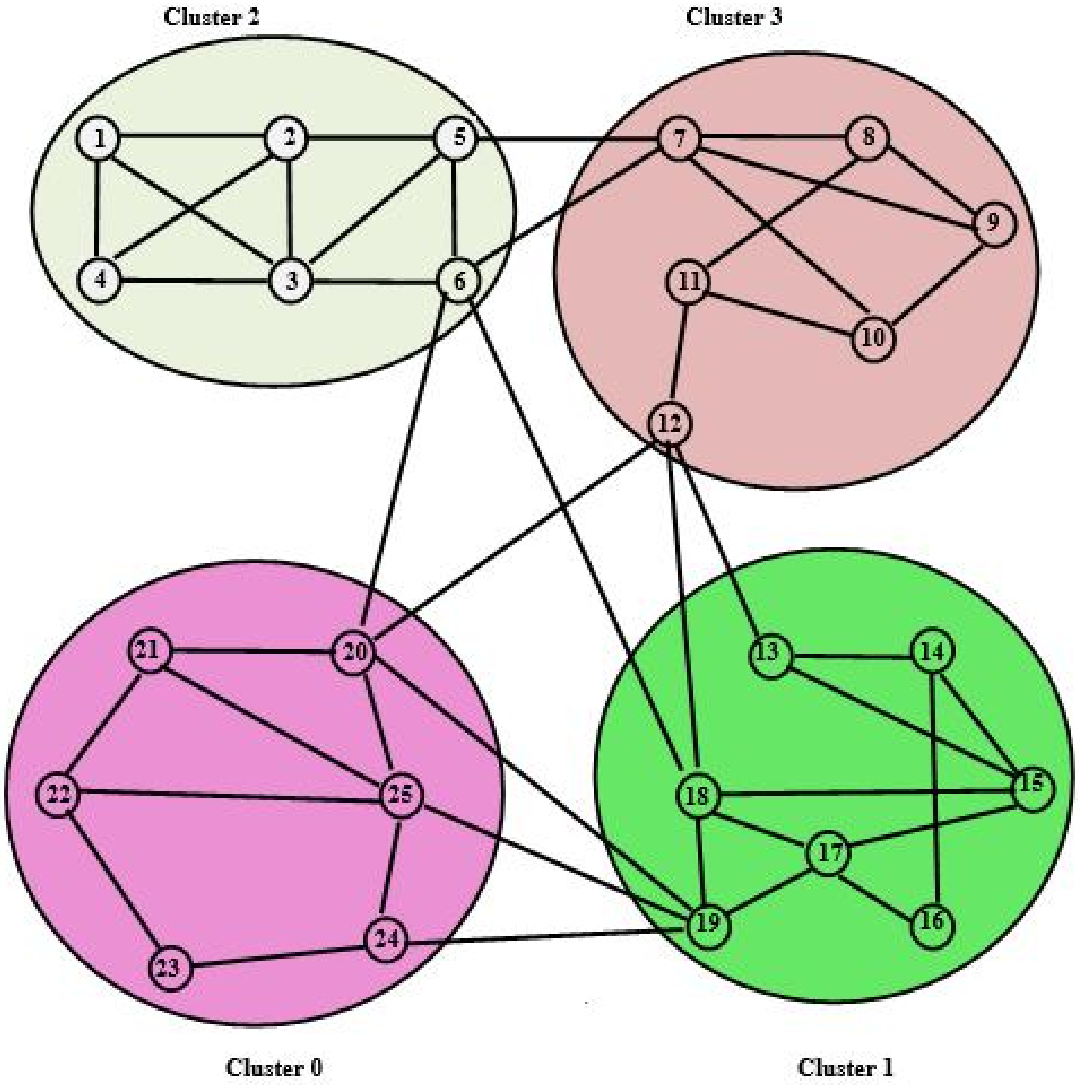


Figure 5.2: Graph *G* partitioned in four clusters.

## 5.3 Graphics Processing Units (GPUs) and CUDA

The Graphics Processing Units (GPUs) have a parallel processing architecture, which allows GPUs to perform multiple calculations at the same time using multi-threading. In 1999, Nvidia introduced the first GPU (GeForce256). The advantages of using the GPUs over CPUs for computation are high performance and usage of less power and lower cost. The interface for GPUs is Compute Unified Device Architecture (CUDA). CUDA is a parallel computing platform created by Nvidia (Corporation, 2010). CUDA is the first language designed by a GPU company to facilitate general-purpose computing on GPUs. CUDA platform is designed to work with C and C++ programming languages. The CUDA platform gives direct access to the GPUs.

In the CPU-GPU heterogeneous environment, the GPU is called the *device* and the CPU to which it is connected is called the *host*. The programs executing on the CPU can access the GPU and data can be transferred from the host memory to the device memory to perform specific tasks.

### 5.3.1 The architecture of GPU

The GPU consists of several Streaming Multiprocessors (SMs) and each multiprocessor contains 8 cores. The cores have access to the shared memory of the specific Streaming Multiprocessor. The Streaming Multiprocessors have access to the global memory (also called device memory). NVIDIA Tesla C1060 Card consists of 30 SMs, 240 GPU cores, 16 KB of shared memory in each of the SM (total of 480 KB of shared memory) and 4 GB of global memory. The architecture of GPU C1060 is shown in Figure 5.3.

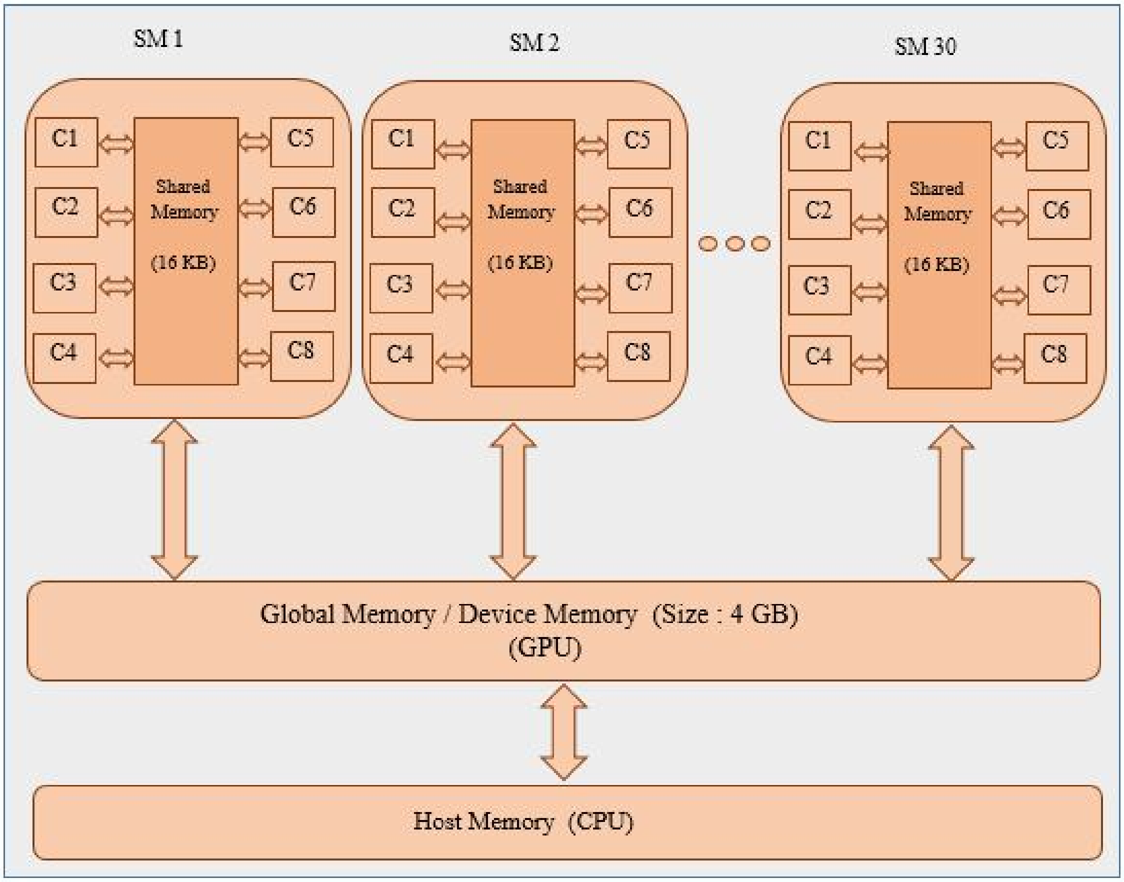


Figure 5.3: GPU architecture (NVIDIA Tesla C1060).

### 5.3.2 CUDA programming model and memory model

The CUDA programming model extends the C programming language. The C language functions are called *kernels* in CUDA. A kernel is defined by using the "__global__" declaration specifier. A CUDA kernel is executed by an array of threads. Each thread has a unique *threadID* to compute memory addresses and to make control decisions. CUDA follows the Single Program Multiple Data (SPMD) model. So, all threads run the same code. In a CUDA program, the sequential code executes in a *host* (CPU) thread and the parallel code executes in many *device* (GPU) threads. The threads are grouped into *blocks*. Blocks can be one-dimensional, two-dimensional, or threedimensional arrays. Blocks can be identified by *blockID*. The blocks are grouped into *grids* and grids can be one-dimensional or two-dimensional arrays. So, the batch of threads that executes a kernel function at device is organized as a grid of thread blocks. The CUDA programming model is shown in Figure 5.4.

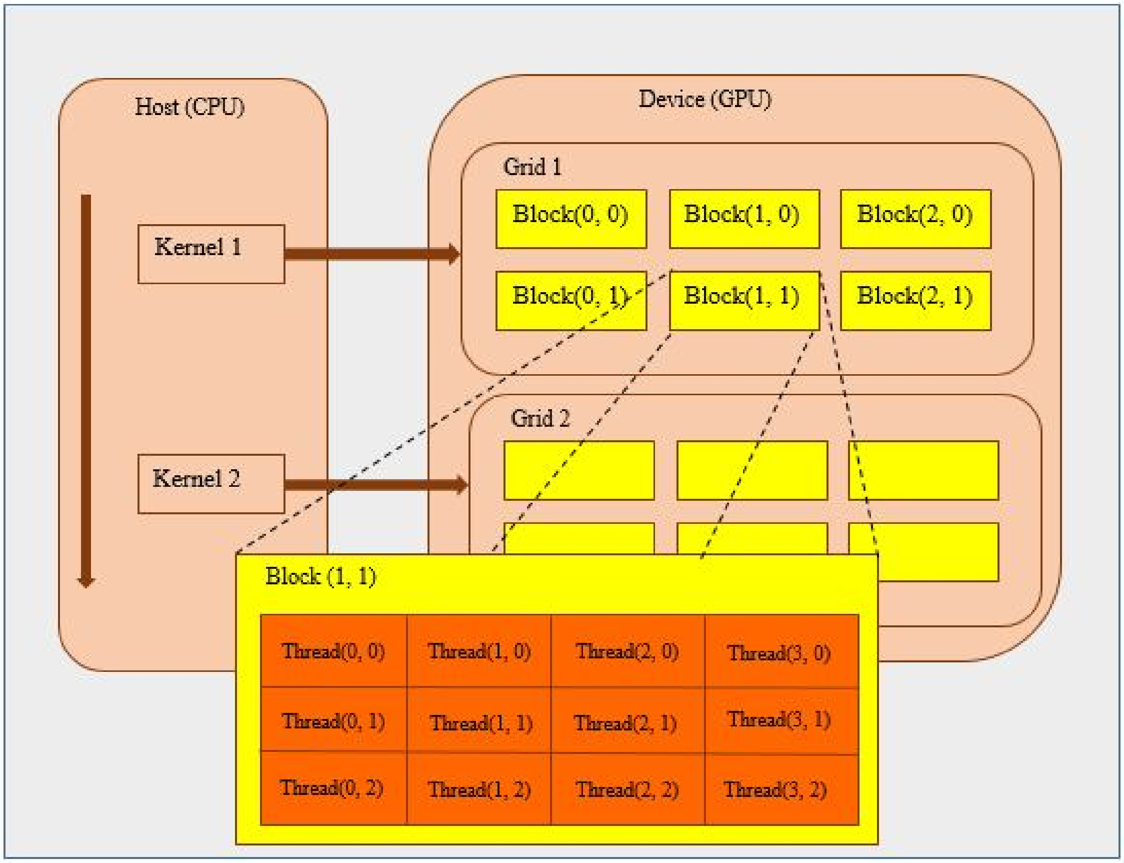


Figure 5.4: CUDA programming model (Corporation, 2010)

On executing a kernel call, the data is transferred from the CPU to the GPU by using memory copy functions and then transferred back to CPU from GPU. Figure 5.5 shows the CUDA memory model. Global memory or device memory is used to transfer data from host to device and then back from device to host. The shared memory is accessed by all the threads within that block. The data stored in the register memory is accessed only by the thread that wrote it.

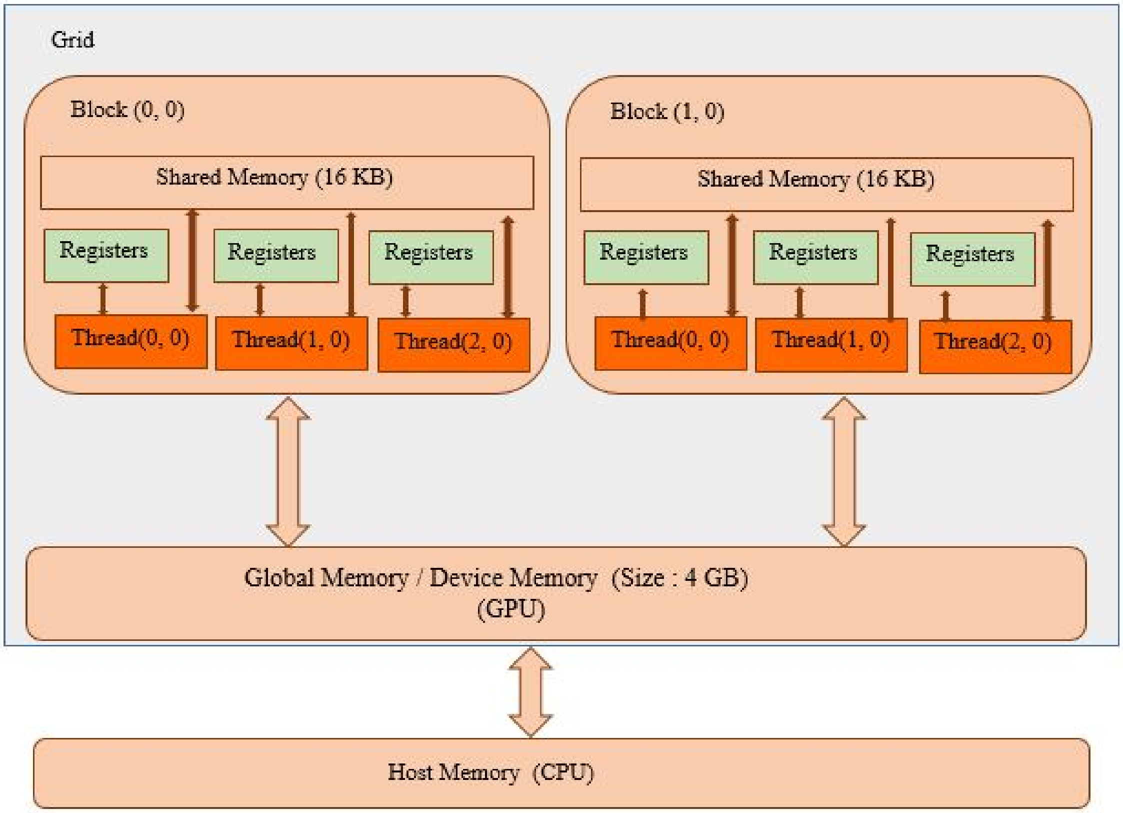


Figure 5.5: CUDA memory model (Corporation, 2010).

## 5.4 Star-Delta Transformation Algorithm using Series and Parallel Reduction

The resistance distance between two nodes of a given network can be calculated by repeated applications of star-delta transformation. Recall from Chapter 4, Star-delta transformation at node *v* of a network *N* is the operation of removing node *v* from *N* and adding a new element between every pair of nodes that are connected to node *v*. To remove a node *v*, we perform series and parallel reductions. Series and parallel reduction along with star-delta are illustrated in Section 4.2 of Chapter 4. See Figure 5.6 for an example.

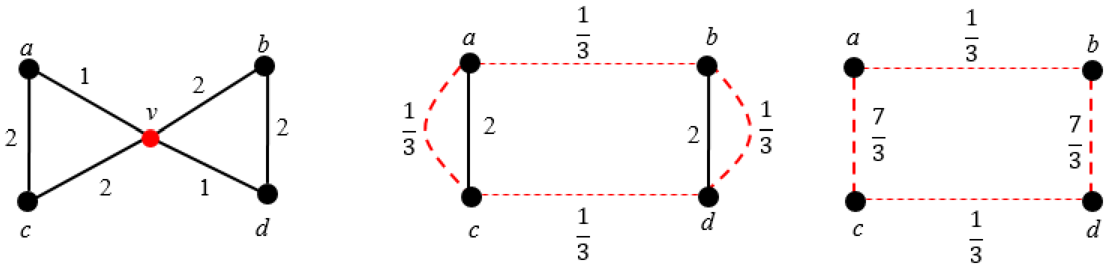


Figure 5.6. Illustration of star-delta transformation and series/parallel reductions.

**Resistance distance algorithm**

For a given network $N$, let $V$ be the set of all nodes in the network. Algorithm 1 given below finds the resistance distance $R$ for nodes $(i, j)$ in $N$.

Algorithm 1**:** Resistance Distance Algorithm

**Step 1**: Set nodes $i$ and $j$ in network $N$.

**Step 2**: Choose the starting node $v$ in $N$ to perform star delta transformation.

**Step 3**: If $v \neq i$ and $v \neq j$, then go to Step 4. Else go to Step 5.

**Step 4**: Perform star-delta transformation on $v$. This will add new resistance elements to all pairs of nodes connected to $v$.

**Step 5**: Remove node $v$.

**Step 6**: Choose next node $v$ to perform star delta transformation if a node $v$ other than $i$ and $j$ is available.

**Step 7**: Repeat Step 3 until all nodes (other than nodes $i$ and $j$ ) have been removed from the network $N$.

**Step 8**: Let the new edge $e$ between nodes $i$ and $j$ have conductance $g$. After performing parallel reduction, the resistance distance $R$ between $i$ and $j$ is

$R = \frac{1}{g}$.

**Step 9**: Choose next pair of nodes ($i$, $j$).

**Step 10**: Repeat Step 1 to Step 9 for all pairs of nodes ($i$, $j$) in *N*.

Figure 5.7 illustrates Algorithm 1 for the node-pair (*a*, *e*). The node picked for star-delta transformation is shown in red color.

## 5.5 Sequential and Parallel Approaches for Resistance Distance Computation

The data structures we have used to store the graph information are *Adjacency List* and *Adjacency Matrix*. For graph partition, we use adjacency list and for finding the resistance distance we use adjacency matrix of the graph. We are using two approaches to find the resistance distance for all pairs of nodes in the graph *G*. In the next two subsections, we explain the sequential and the parallel approach for resistance distance.

### 5.5.1 Sequential approach for resistance distance

For finding the resistance distance using the star-delta transformation procedure, we need to update the given adjacency matrix *A*. The sequential approach for finding the resistance distance is given in Algorithm 2. The input for this algorithm is the adjacency matrix *A* of graph *G* and the output is the resistance distance matrix *R* for all pairs of nodes in *G*. In Algorithm 2, first we get the number of nodes *n* in *G*. Then for all pairs of nodes, we calculate the sum of elements of all the rows in adjacency

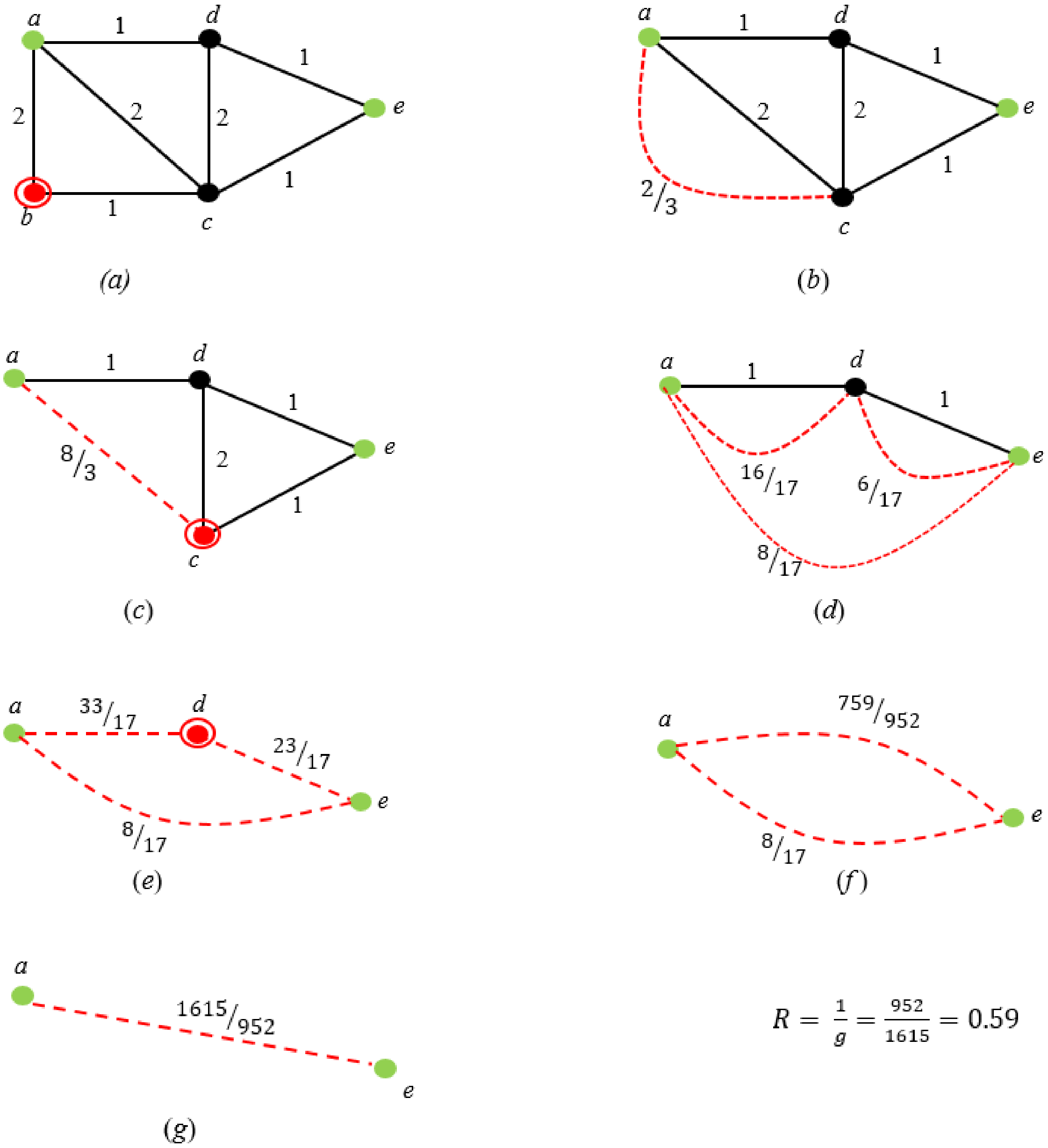


Figure 5.7: Illustration of star-delta transformation procedure algorithm.

matrix *A* and store them in *rowSumArray*[*n*]. Then we find the non-zero columns of a row in adjacency matrix and store them in *jRowArray*[*n*]. The information of nodes given in *jRowArray* helps in updating the adjacency matrix *A*. Then we set all the elements of the processed row *i* and column *i* to zero in adjacency matrix.

Algorithm 2: Sequential Algorithm for Resistance Distance Calculation

```
Input : Adjacency Matrix A of graph G.
Output: Resistance Distance R for all pairs of nodes in graph G.
begin
  Get n (number of nodes in G);
  A[n][n] ← Adjacency Matrix (G);
  for $i = 0$ to $n - 1$ do
      for $j = 0$ to $n - 1$ do
          rowSumArray[i] ← rowSumArray[i] + A[i][j];
          count ← 0;
          for $iRow = 0$ to $n - 1$ do
              for $jRow = 0$ to $n - 1$ do
                  if A[iRow][jRow] != 0 then
                      jRowArray[count] ← jRow;
                      count ← count + 1;
                  end if
              end for
              for $index1 = 0$ to count do
                  jUpdate1 ← jRowArray[index1] ;
                  for $index2 = index1 + 1$ to count do
                      jUpdate2 ← jRowArray[index2];
                      addition ← $\frac{A[iRow][jUpdate1]*A[iRow][jUpdate2]}{rowSumArray[iRow]}$ ;
                      $A[jUpdate1][jUpdate2]$←$A[jUpdate1][jUpdate2]$ +
                                                  $addition$;
                  end for
                  $A[iRow][jUpdate1]$← 0;
                  $A[jUpate1][iRow]$← 0;
              end for
              count ← 0;
              rowSumArray[i] ← $rowSumArray[i] + A[i][j]$;
```

```
                end for
                for $rIndex1 = 0$ to $n - 1$ do
                    for $rIndex2 = 0$ to $n - 1$ do
                        if $A[rIndex1][rIndex2]! = 0$ then
                            $R[rIndex1][rIndex2] \leftarrow \frac{1}{A[rIndex1][rIndex2]}$;
                        end if
                    end for
                end for
            end for
    end for
     Output ← $R[n][n]$;
end
```

Updating adjacency matrix completes the star-delta transformation at the selected nodes. Once the adjacency matrix is updated, we again calculate the new sum of the rows and update *rowSumArray*[*n*]. Then we calculate the resistance distance by taking the reciprocal of the updated adjacency matrix and storing them in resistance distance matrix *R*.

### 5.5.2 Parallel approach for resistance distance

For the parallel approach, we use CUDA parallel programming. Recall from Section 5.2 that the sequential part of the code is executed on the CPU (*host*), and the parallel parts are executed on the GPU (*device*). Algorithms 3 and 4 explain the parallel parts that are executed on the device. These are similar to Algorithms 1 and 2 incorporating certain features required for parallel execution. Algorithm 3 is the device code, *kernel_rowSum*(*A*, *rowSumArray*) function, for calculating the sum of elements of all the rows in adjacency matrix *A* and storing them in *rowSumArray*. The __syncthreads() function is used to coordinate the threads. This function works as a block level synchronization barrier and it makes all threads stop at a certain point in the kernel before moving enmasse.

Algorithm 3: *kernel_rowSum* – device code

```
Procedure kernel_rowSum (A, rowSumArray)
    Get n (number of nodes);
    $i \leftarrow blockDim.x * blockIdx.x + threadIdx.x$;
    if $i < n$ then
        rowSumArray[$i$] ← 0;
        for $j = 0$ to $n$ do
            $rowSumArray[i] \leftarrow rowSumArray[i] + A[i * n + j]$;
        end for
         __syncthreads();
    end if
end procedure
```

Algorithm 4 is the device code, *kernel_updateMatrix*(*A*, *rowSumArray, jRowArray, irow, count*) function, for updating the adjacency matrix *A*. Here *rowSumArray* is the array of sum of rows of adjacency matrix *A*, *jRowArray* is the array that holds the information of nodes to be updating in the adjacency matrix.

Algorithm 4: *kernel_updateMatrix* – device code

```
Procedure kernel_updateMatrix (A, rowSumArray, jRowArray, irow, count)
    Get n (number of nodes);
    $i \leftarrow blockDim.y * blockIdx.y + threadIdx.y$;
    $j \leftarrow blockDim.x * blockIdx.x + threadIdx.x$;
    if $i < n$ and $j < n$ then
        for $index1 = 0$ to $n - 1$ do
```

```
            **for** *index2* = 0 to *k* **do**
                *jUpdate*1 ← *jRowArray*[*index2*];
                **if** *index*1 = *jUpdate*1 **then**
                    **for** $index3 = index1 + 1$ to $n - 1$ **do**
                        **for** *index*4 = 0 to *k* **do**
                            *jUpdate2* ← *jRowArray*[*index*4];
                            **if** *index*3 = *jUpdate*2 **then**
                                addition = $\frac{A[iRow*n+jUpdate1]*A[iRow*n+jUpdate2]}{rowSumArray[iRow]}$
                                $A[index1 * n + index3]$← $A[index1 * n + index3] + addition$;
                                $A[index3 * n + index1]$← $A[index3 * n + index1] + addition$;
                            **end if**
                        **end for**
                        __syncthreads( );
                    **end for**
                    $A[irow * n + index1] = 0$;
                    $A[index1 * n + irow = 0$;
                **end if**
            **end for**
            __syncthreads();
        **end for**
    **end if**
**end procedure**
```

The parallel approach for finding the resistance distance is given in Algorithm 5. The input for this algorithm is the adjacency matrix *A* of graph *G* and the output is the resistance distance matrix *R* for all pair of nodes in *G*. In Algorithm 5, *GPUMalloc*( ) function requests the array on the device's global memory and *GPUFree*() function frees the array from the device global memory. *MemcpyHostToDevice*( ) function transfers data from host memory to device memory and *MemcpyDeviceToHost*( ) function transfers data back to host memory from device memory.

To call the *kernel* functions from the device, we declare *blocksPerGrid* and *threadsPerBlock*. *blocksPerGrid* is the number of blocks we want to run on processors in parallel and *threadsPerBlock* is the number of threads we want to activate per block. We call the *kernel_rowSum* function given in Algorithm 3 to calculate the sum of the rows of adjacency matrix *A*. Then we call *kernel_updateMatrix* function given in Algorithm 4 to update the entries of the adjacency matrix. Calculate the resistance distance by taking the reciprocal of the updated adjacency matrix elements and storing them in resistance distance matrix *R*.

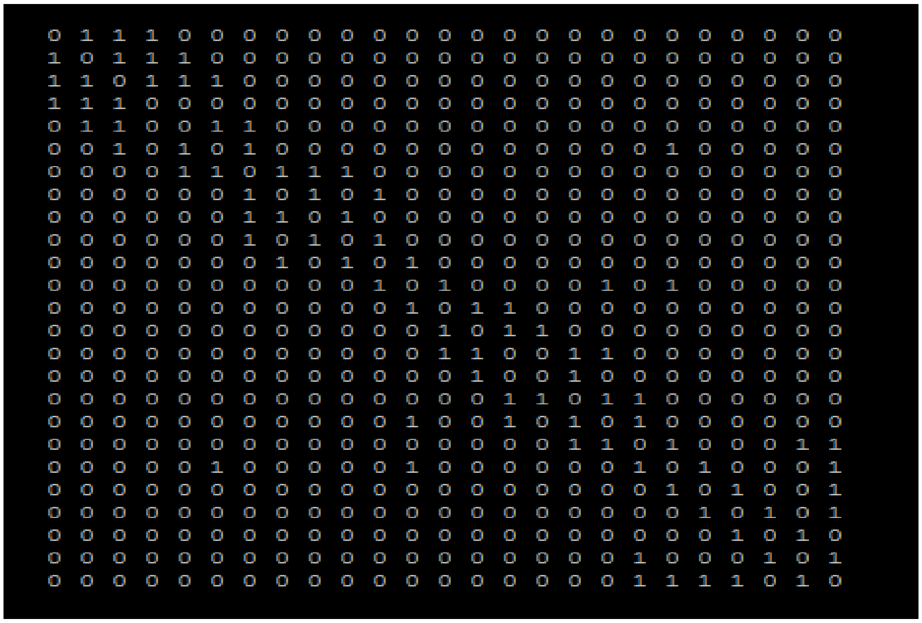
Algorithm 5: Parallel Algorithm for Resistance Distance Calculation

```
Input : Adjacency Matrix A of graph G.
Output: Resistance Distance R for all pairs of nodes in graph G.
begin
   Get n (number of nodes in G);
   A[n][n] ← Adjacency Matrix (G);
   for $i = 0$ to $n - 1$ do
        for $j = i + 1$ to $n - 1$ do
     // Call $kernel\_rowSum$ function to add the elements of rows of adjacency matrix
            GPUMalloc( );
            MemcpyHostToDevice( );
            $kernel\_rowSum$<<<$blocksPerGrid, threadsPerBlock$>>>(A,
                rowSumArray);
            MemcpyDeviceToHost( );
            GPUFree( );
            count ← 0;
```

```
for $iRow = 0$ to $n - 1$ do
    for $jRow = 0$ to $n - 1$ do
        if A[iRow][jRow] != 0 then
            jRowArray[count] ← jRow;
            count ← count + 1;
        end if
    end for
    // Call kernel_updateMatrix function to update the adjacency matrix.
    GPUMalloc( );
    MemcpyHostToDevice( );
  kernel_updateMatrix<<<blocksPerGrid, threadsPerBlock>>>(A,
            rowSumArray, jRowArray, irow, count);
    MemcpyDeviceToHost( );
    GPUFree( );
    count ← 0;
  // Call kernel_rowSum function to add the rows of updated adjaceny matrix
      GPUMalloc( );
      MemcpyHostToDevice( );
      kernel_rowSum<<<blocksPerGrid, threadsPerBlock>>>(A,
                rowSumArray);
      MemcpyDeviceToHost( );
      GPUFree( );
end for
for $rIndex1 = 0$ to $n - 1$ do
    for $rIndex2 = 0$ to $n - 1$ do
        if $A[rIndex1][rIndex2]! = 0$ then
            $R[rIndex1][rIndex2]$ ← $\frac{1}{A[rIndex1][rIndex2]}$;
        end if
    end for
end for
```

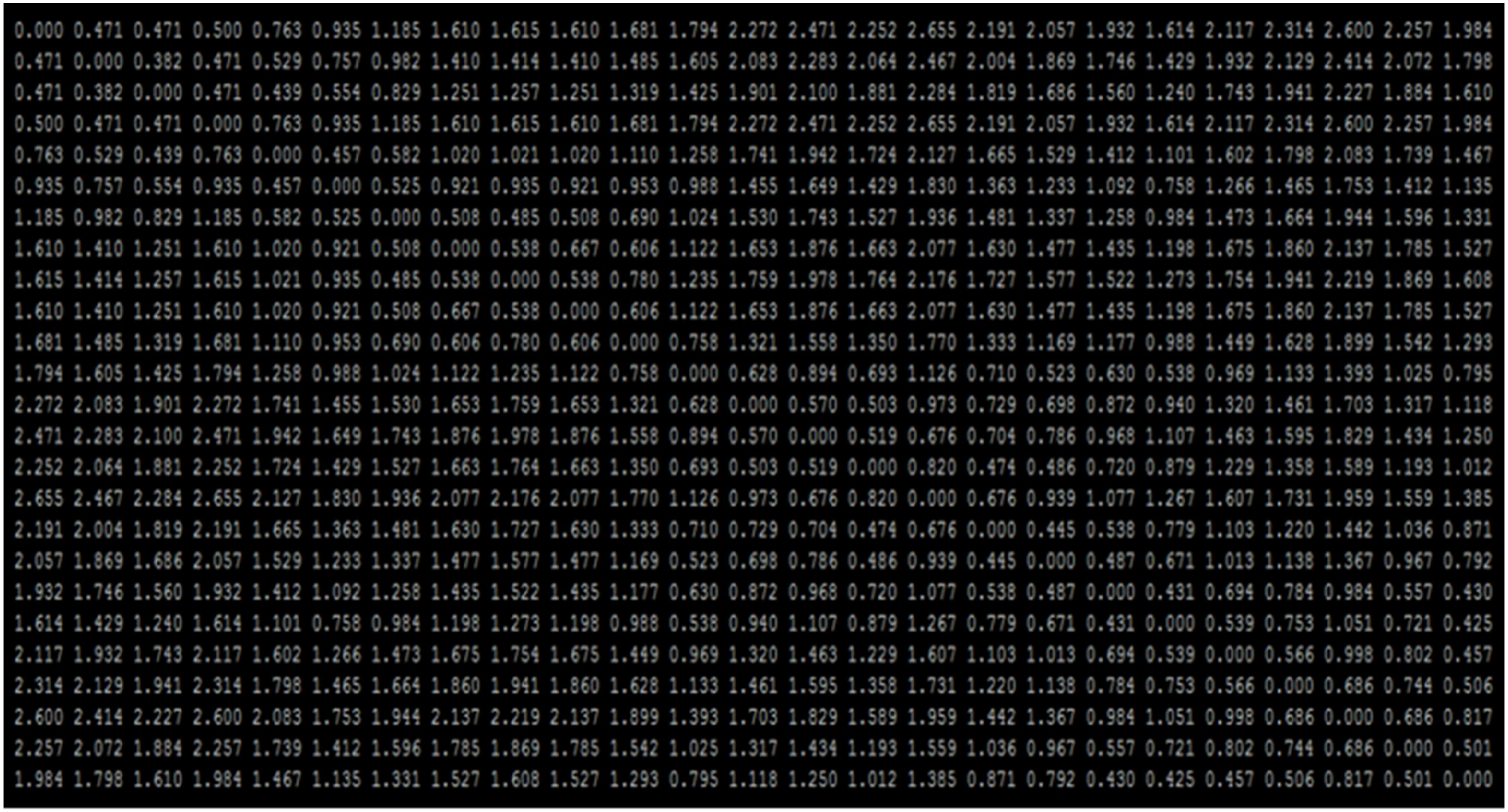
```
        end for
    end for
    Output ← R[n][n];
End
```

We are getting the same output resistance distance matrix *R* from the sequential approach Algorithm 2 and parallel approach Algorithm 5. Figure 5.8 shows the input adjacency matrix *A* and Figure 5.9 shows the output resistance distance matrix *R* of *G* given in Figure 5.1(*a*).



```
0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  1  1  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0
0  0  0  0  1  1  0  1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  1  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  1  0  1  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  1  0  1  1  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  1  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0  1  1  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  1  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  1  0  1  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0  0  1  1
0  0  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0  1  0  1  0  0  0  1
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  1
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  1
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  1  0  1
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  1  1  0  1  0
```

Figure 5.8: Adjacency matrix *A* of *G* given in Figure 5.1(*a*).



```
0.000 0.471 0.471 0.500 0.763 0.935 1.185 1.610 1.615 1.610 1.681 1.794 2.272 2.471 2.252 2.655 2.191 2.057 1.932 1.614 2.117 2.314 2.600 2.257 1.984
0.471 0.000 0.382 0.471 0.529 0.757 0.982 1.410 1.414 1.410 1.485 1.605 2.083 2.283 2.064 2.467 2.004 1.869 1.746 1.429 1.932 2.129 2.414 2.072 1.798
0.471 0.382 0.000 0.471 0.439 0.554 0.829 1.251 1.257 1.251 1.319 1.425 1.901 2.100 1.881 2.284 1.819 1.686 1.560 1.240 1.743 1.941 2.227 1.884 1.610
0.500 0.471 0.471 0.000 0.763 0.935 1.185 1.610 1.615 1.610 1.681 1.794 2.272 2.471 2.252 2.655 2.191 2.057 1.932 1.614 2.117 2.314 2.600 2.257 1.984
0.763 0.529 0.439 0.763 0.000 0.457 0.582 1.020 1.021 1.020 1.110 1.258 1.741 1.942 1.724 2.127 1.665 1.529 1.412 1.101 1.602 1.798 2.083 1.739 1.467
0.935 0.757 0.554 0.935 0.457 0.000 0.525 0.921 0.935 0.921 0.953 0.988 1.455 1.649 1.429 1.830 1.363 1.233 1.092 0.758 1.266 1.465 1.753 1.412 1.135
1.185 0.982 0.829 1.185 0.582 0.525 0.000 0.508 0.485 0.508 0.690 1.024 1.530 1.743 1.527 1.936 1.481 1.337 1.258 0.984 1.473 1.664 1.944 1.596 1.331
1.610 1.410 1.251 1.610 1.020 0.921 0.508 0.000 0.538 0.667 0.606 1.122 1.653 1.876 1.663 2.077 1.630 1.477 1.435 1.198 1.675 1.860 2.137 1.785 1.527
1.615 1.414 1.257 1.615 1.021 0.935 0.485 0.538 0.000 0.538 0.780 1.235 1.759 1.978 1.764 2.176 1.727 1.577 1.522 1.273 1.754 1.941 2.219 1.869 1.608
1.610 1.410 1.251 1.610 1.020 0.921 0.508 0.667 0.538 0.000 0.606 1.122 1.653 1.876 1.663 2.077 1.630 1.477 1.435 1.198 1.675 1.860 2.137 1.785 1.527
1.681 1.485 1.319 1.681 1.110 0.953 0.690 0.606 0.780 0.606 0.000 0.758 1.321 1.558 1.350 1.770 1.333 1.169 1.177 0.988 1.449 1.628 1.899 1.542 1.293
1.794 1.605 1.425 1.794 1.258 0.988 1.024 1.122 1.235 1.122 0.758 0.000 0.628 0.894 0.693 1.126 0.710 0.523 0.630 0.538 0.969 1.133 1.393 1.025 0.795
2.272 2.083 1.901 2.272 1.741 1.455 1.530 1.653 1.759 1.653 1.321 0.628 0.000 0.570 0.503 0.973 0.729 0.698 0.872 0.940 1.320 1.461 1.703 1.317 1.118
2.471 2.283 2.100 2.471 1.942 1.649 1.743 1.876 1.978 1.876 1.558 0.894 0.570 0.000 0.519 0.676 0.704 0.786 0.968 1.107 1.463 1.595 1.829 1.434 1.250
2.252 2.064 1.881 2.252 1.724 1.429 1.527 1.663 1.764 1.663 1.350 0.693 0.503 0.519 0.000 0.820 0.474 0.486 0.720 0.879 1.229 1.358 1.589 1.193 1.012
2.655 2.467 2.284 2.655 2.127 1.830 1.936 2.077 2.176 2.077 1.770 1.126 0.973 0.676 0.820 0.000 0.676 0.939 1.077 1.267 1.607 1.731 1.959 1.559 1.385
2.191 2.004 1.819 2.191 1.665 1.363 1.481 1.630 1.727 1.630 1.333 0.710 0.729 0.704 0.474 0.676 0.000 0.445 0.538 0.779 1.103 1.220 1.442 1.036 0.871
2.057 1.869 1.686 2.057 1.529 1.233 1.337 1.477 1.577 1.477 1.169 0.523 0.698 0.786 0.486 0.939 0.445 0.000 0.487 0.671 1.013 1.138 1.367 0.967 0.792
1.932 1.746 1.560 1.932 1.412 1.092 1.258 1.435 1.522 1.435 1.177 0.630 0.872 0.968 0.720 1.077 0.538 0.487 0.000 0.431 0.694 0.784 0.984 0.557 0.430
1.614 1.429 1.240 1.614 1.101 0.758 0.984 1.198 1.273 1.198 0.988 0.538 0.940 1.107 0.879 1.267 0.779 0.671 0.431 0.000 0.539 0.753 1.051 0.721 0.425
2.117 1.932 1.743 2.117 1.602 1.266 1.473 1.675 1.754 1.675 1.449 0.969 1.320 1.463 1.229 1.607 1.103 1.013 0.694 0.539 0.000 0.566 0.998 0.802 0.457
2.314 2.129 1.941 2.314 1.798 1.465 1.664 1.860 1.941 1.860 1.628 1.133 1.461 1.595 1.358 1.731 1.220 1.138 0.784 0.753 0.566 0.000 0.686 0.744 0.506
2.600 2.414 2.227 2.600 2.083 1.753 1.944 2.137 2.219 2.137 1.899 1.393 1.703 1.829 1.589 1.959 1.442 1.367 0.984 1.051 0.998 0.686 0.000 0.686 0.817
2.257 2.072 1.884 2.257 1.739 1.412 1.596 1.785 1.869 1.785 1.542 1.025 1.317 1.434 1.193 1.559 1.036 0.967 0.557 0.721 0.802 0.744 0.686 0.000 0.501
1.984 1.798 1.610 1.984 1.467 1.135 1.331 1.527 1.608 1.527 1.293 0.795 1.118 1.250 1.012 1.385 0.871 0.792 0.430 0.425 0.457 0.506 0.817 0.501 0.000
```

Figure 5.9. Resistance Distance Matrix *R* of graph *G* given in Figure 5.1(*a*).

## 5.6 Three-Step Approximation Algorithm for Resistance Distance Calculation

In this section, our main objective is to introduce a three-step approximation algorithm to calculate resistance distance between all pairs of nodes of a network. Algorithm 6 finds the resistance distance $R_{approx}$ using the paths in a network. First the network is partitioned into clusters. Figure 5.10 shows the boundary nodes network $B'$ having weights on the edges (dashed red color edges) of boundary nodes of each cluster (Illustrate Step 1 and Step 3 of Algorithm 6). The relationship between the resistance distance *R* we are getting from Algorithm 2 (Algorithm 5 for parallel) and the resistance distance $R_{approx}$ from Algorithm 6 is

$$R_{approx} \geq R.$$

Figure 5.11 shows the output resistance distance matrix $R_{approx}$ of graph *G* given in Figure 5.2.

Algorithm 6: Three-Step Approximation Algorithm for Resistance Distance Calculation

**Step 1:** Find the boundary nodes of each cluster in network *N*. These are the nodes that connect inter-cluster edges. See Figure 5.2.

**Step 2:** Get the adjacency matrix $A_{cluster}$ for each of the clusters in the network.

**Step 3:** Find the weight on the edges of boundary nodes of each cluster by using Algorithm 2 (for sequential approach) or Algorithm 5 (for parallel approach).

**Step 4:** Get the adjacency matrix $A_{boundary}$ for network $B'$ of boundary nodes.

**Step 5:** Get the resistance distance matrix $R_{boundary}$ for each pair of boundary nodes in $B'$ by using Algorithm 2 or Algorithm 5. The input for Algorithm 2 and Algorithm 5 is adjacency matrix $A_{boundary}$.

**Step 6:** Get the adjacency matrix *A* of the network *N*.

**Step 7:** To find the resistance distance *R* for each pair of nodes (*i*, and *j*) in *N*, go to

Step 8.

**Step 8:** Set nodes *i* and *j*.

**Step 9:** If nodes *i* and *j* are in the same cluster and both are non-boundary nodes then get the resistance distance by using Algorithm 2 or Algorithm 5. The input adjacency is matrix *A*. Go to Step 19.

**Step 10:** If nodes *i* and *j* are in the same cluster but *i* is a non-boundary node of the cluster and *j* is a boundary node of the cluster then get the resistance distance $R_{cluster}[i][j]$ by using Algorithm 2 or Algorithm 5. Go to Step 19.

**Step 11:** If nodes *i* and *j* are in different clusters and both are non-boundary nodes of those clusters then go to Step 12.

**Step 12:** If node *i* is a non-boundary node in *clusterA* and node *j* is non-boundary node in *clusterB* then go to Step 13 through Step 16 and find the resistance distance $R_{approx}[i][j]$.

**Step 13:** Find the resistance distance $R_{clusterA}[i][k]$ from node *i* to each boundary node *k* of *clusterA*, by using Algorithm 2 or Algorithm 5. The input adjacency is matrix $A_{clusterA}$.

**Step 14:** Find the resistance distance $R_{clusterB}[k][j]$ from node *j* to boundary node *k* of *clusterB*, by using Algorithm 2 or Algorithm 5. The input adjacency matrix is $A_{clusterB}$.

**Step 15:** Find the resistance distance $R_{boundary}[k][l]$ from boundary node *k* of *clusterA* to boundary node *l* of *clusterB*, by using Algorithm 2 or Algorithm 5. The input adjacency matrix is $A_{boundary}$.

**Step 16:** Set $R_{approx}[i][j] \leftarrow$ minimum resistance distance from *i* to *j* using paths of length of 3, containing the boundary nodes of *clusterA* and *clusterB*.

Go to Step 19.

**Step 17:** If node *i* is a non-boundary node in *clusterA* and node *j* is a boundary node in *clusterB* then repeat Step 13 and Step 15 and find resistance distance.

Set $R_{approx}[i][j] \leftarrow$ the minimum resistance distance along paths of length 2, containing only the boundary nodes of *clusterA*.

Go to Step 19.

```
Step 18: If node $i$ is a boundary node in clusterA and node $j$ is a non-boundary node in
         clusterB then repeat Step 14 and Step 15 and find resistance distance
         $R_{diversePath}[i][j]$.
         Set $R_{approx}[i][j] \leftarrow$ the minimum resistance distance of paths of length 2 from
                 $i$ to $j$, containing only the boundary nodes of ClusterB.
         Go to Step 19.
Step 19: Choose the next pair of nodes ($i$, $j$). Go to Step 8.
Step 20: Stop when all pairs of nodes have been considered.
```

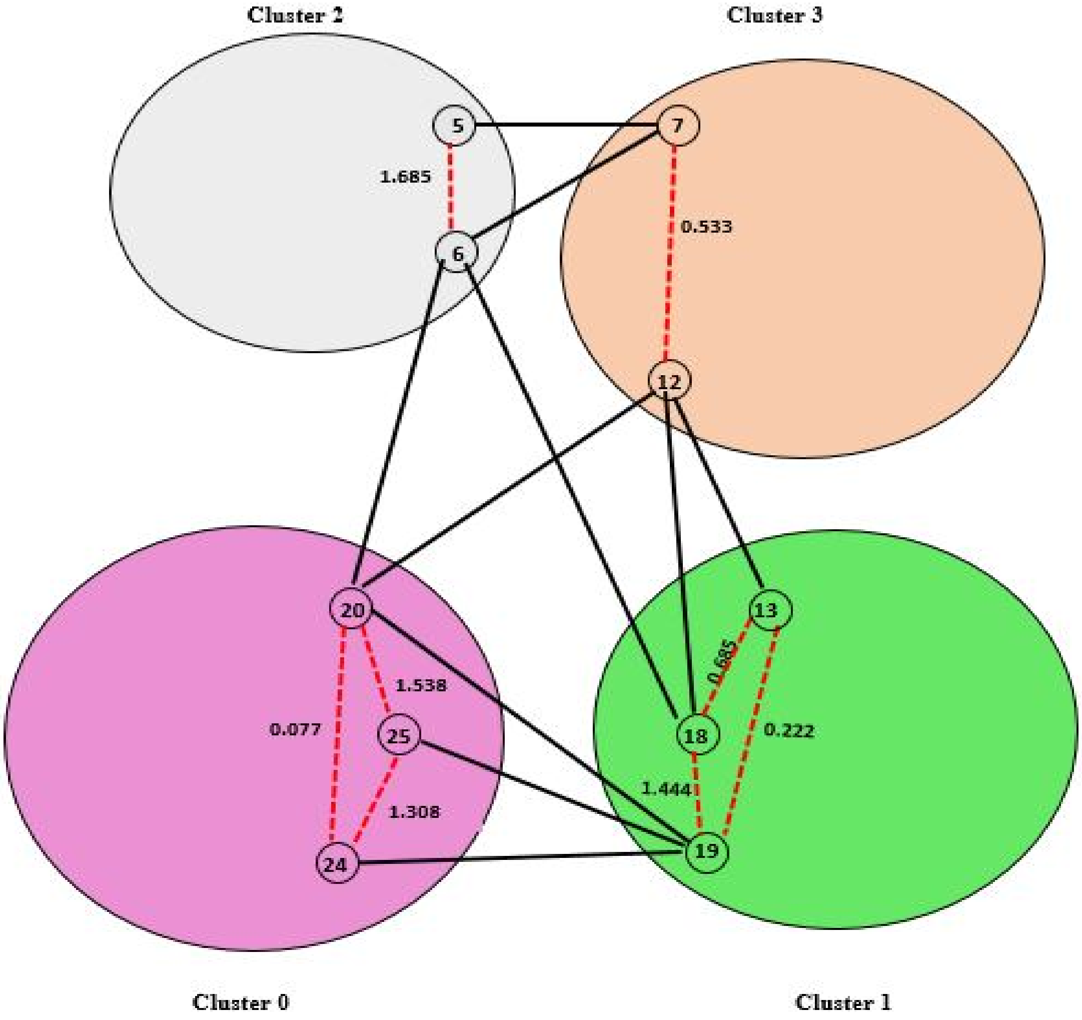
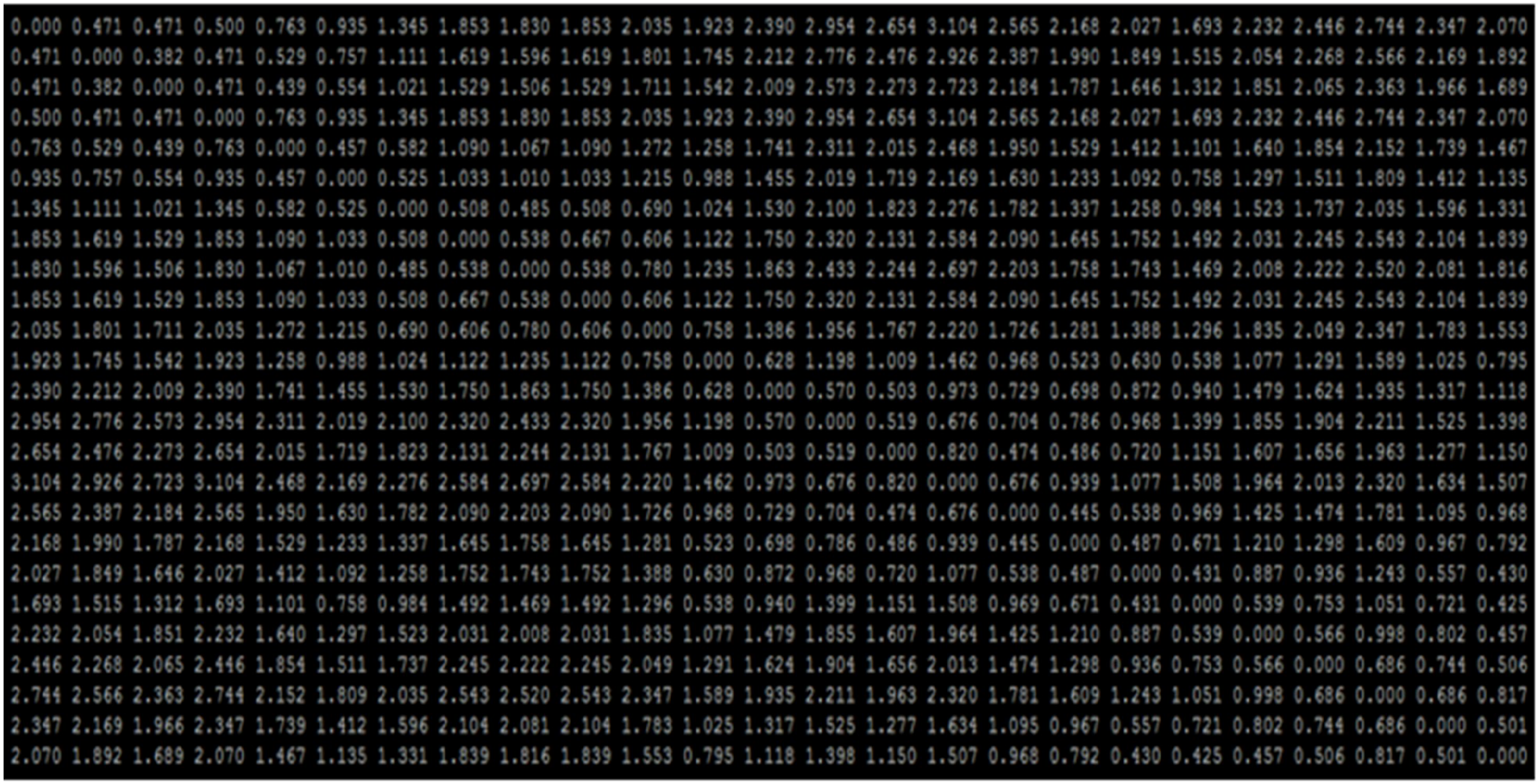


Figure 5.10**:** Boundary node network *B'* of graph *G* in Figure 5.2.



```
0.000 0.471 0.471 0.500 0.763 0.935 1.345 1.853 1.830 1.853 2.035 1.923 2.390 2.954 2.654 3.104 2.565 2.168 2.027 1.693 2.232 2.446 2.744 2.347 2.070
0.471 0.000 0.382 0.471 0.529 0.757 1.111 1.619 1.596 1.619 1.801 1.745 2.212 2.776 2.476 2.926 2.387 1.990 1.849 1.515 2.054 2.268 2.566 2.169 1.892
0.471 0.382 0.000 0.471 0.439 0.554 1.021 1.529 1.506 1.529 1.711 1.542 2.009 2.573 2.273 2.723 2.184 1.787 1.646 1.312 1.851 2.065 2.363 1.966 1.689
0.500 0.471 0.471 0.000 0.763 0.935 1.345 1.853 1.830 1.853 2.035 1.923 2.390 2.954 2.654 3.104 2.565 2.168 2.027 1.693 2.232 2.446 2.744 2.347 2.070
0.763 0.529 0.439 0.763 0.000 0.457 0.582 1.090 1.067 1.090 1.272 1.258 1.741 2.311 2.015 2.468 1.950 1.529 1.412 1.101 1.640 1.854 2.152 1.739 1.467
0.935 0.757 0.554 0.935 0.457 0.000 0.525 1.033 1.010 1.033 1.215 0.988 1.455 2.019 1.719 2.169 1.630 1.233 1.092 0.758 1.297 1.511 1.809 1.412 1.135
1.345 1.111 1.021 1.345 0.582 0.525 0.000 0.508 0.485 0.508 0.690 1.024 1.530 2.100 1.823 2.276 1.782 1.337 1.258 0.984 1.523 1.737 2.035 1.596 1.331
1.853 1.619 1.529 1.853 1.090 1.033 0.508 0.000 0.538 0.667 0.606 1.122 1.750 2.320 2.131 2.584 2.090 1.645 1.752 1.492 2.031 2.245 2.543 2.104 1.839
1.830 1.596 1.506 1.830 1.067 1.010 0.485 0.538 0.000 0.538 0.780 1.235 1.863 2.433 2.244 2.697 2.203 1.758 1.743 1.469 2.008 2.222 2.520 2.081 1.816
1.853 1.619 1.529 1.853 1.090 1.033 0.508 0.667 0.538 0.000 0.606 1.122 1.750 2.320 2.131 2.584 2.090 1.645 1.752 1.492 2.031 2.245 2.543 2.104 1.839
2.035 1.801 1.711 2.035 1.272 1.215 0.690 0.606 0.780 0.606 0.000 0.758 1.386 1.956 1.767 2.220 1.726 1.281 1.388 1.296 1.835 2.049 2.347 1.783 1.553
1.923 1.745 1.542 1.923 1.258 0.988 1.024 1.122 1.235 1.122 0.758 0.000 0.628 1.198 1.009 1.462 0.968 0.523 0.630 0.538 1.077 1.291 1.589 1.025 0.795
2.390 2.212 2.009 2.390 1.741 1.455 1.530 1.750 1.863 1.750 1.386 0.628 0.000 0.570 0.503 0.973 0.729 0.698 0.872 0.940 1.479 1.624 1.935 1.317 1.118
2.954 2.776 2.573 2.954 2.311 2.019 2.100 2.320 2.433 2.320 1.956 1.198 0.570 0.000 0.519 0.676 0.704 0.786 0.968 1.399 1.855 1.904 2.211 1.525 1.398
2.654 2.476 2.273 2.654 2.015 1.719 1.823 2.131 2.244 2.131 1.767 1.009 0.503 0.519 0.000 0.820 0.474 0.486 0.720 1.151 1.607 1.656 1.963 1.277 1.150
3.104 2.926 2.723 3.104 2.468 2.169 2.276 2.584 2.697 2.584 2.220 1.462 0.973 0.676 0.820 0.000 0.676 0.939 1.077 1.508 1.964 2.013 2.320 1.634 1.507
2.565 2.387 2.184 2.565 1.950 1.630 1.782 2.090 2.203 2.090 1.726 0.968 0.729 0.704 0.474 0.676 0.000 0.445 0.538 0.969 1.425 1.474 1.781 1.095 0.968
2.168 1.990 1.787 2.168 1.529 1.233 1.337 1.645 1.758 1.645 1.281 0.523 0.698 0.786 0.486 0.939 0.445 0.000 0.487 0.671 1.210 1.298 1.609 0.967 0.792
2.027 1.849 1.646 2.027 1.412 1.092 1.258 1.752 1.743 1.752 1.388 0.630 0.872 0.968 0.720 1.077 0.538 0.487 0.000 0.431 0.887 0.936 1.243 0.557 0.430
1.693 1.515 1.312 1.693 1.101 0.758 0.984 1.492 1.469 1.492 1.296 0.538 0.940 1.399 1.151 1.508 0.969 0.671 0.431 0.000 0.539 0.753 1.051 0.721 0.425
2.232 2.054 1.851 2.232 1.640 1.297 1.523 2.031 2.008 2.031 1.835 1.077 1.479 1.855 1.607 1.964 1.425 1.210 0.887 0.539 0.000 0.566 0.998 0.802 0.457
2.446 2.268 2.065 2.446 1.854 1.511 1.737 2.245 2.222 2.245 2.049 1.291 1.624 1.904 1.656 2.013 1.474 1.298 0.936 0.753 0.566 0.000 0.686 0.744 0.506
2.744 2.566 2.363 2.744 2.152 1.809 2.035 2.543 2.520 2.543 2.347 1.589 1.935 2.211 1.963 2.320 1.781 1.609 1.243 1.051 0.998 0.686 0.000 0.686 0.817
2.347 2.169 1.966 2.347 1.739 1.412 1.596 2.104 2.081 2.104 1.783 1.025 1.317 1.525 1.277 1.634 1.095 0.967 0.557 0.721 0.802 0.744 0.686 0.000 0.501
2.070 1.892 1.689 2.070 1.467 1.135 1.331 1.839 1.816 1.839 1.553 0.795 1.118 1.398 1.150 1.507 0.968 0.792 0.430 0.425 0.457 0.506 0.817 0.501 0.000
```

Figure 5.11. Resistance Distance Matrix $R_{approx}$ of graph $G$ given in Figure 5.2.

## 5.7 Experiment Results

The software used for graph partitioning for our experiments is Metis 5.1.0, as discussed in Section 5.2. The programs are written in C and CUDA (toolkit 5.5) and compiled using the GCC v4.8.2 and nvcc compilers on a Linux x86_64 version 3.10.0. The sequential Algorithm 3 is implemented using CPU and the parallel Algorithm 5 is implemented using both CPU and GPU. The CPU implementation is performed using a single thread. The CPU used for experiments consists of quad-core 2.27 GHz Intel Xeon processors with 12GB of memory. The GPU used for experiments is Nvidia C1060 card with 240 GPU cores and 4GB of memory.

The experiments have been performed on datasets (Johnson, et al., 1989) for graph sizes ranging from 25 to 500 nodes. The graphs considered for experiments in this chapter are graphExample (25 nodes, 44 edges), G124 (124 nodes, 318 edges), G250 (250 nodes, 1283 edges) and G500 (500 nodes, 5120 edges). The timing for the sequential (Algorithm 2) and parallel (Algorithm 5) implementation is shown in Table 3. The time for parallel implementation is less than the sequential implementation. The time for parallel implementation for graph G250 for various number of processors is shown in Table 4. The time is more for parallel implementation if the number of processors are less.

The quality of performance of three-step approximation algorithm 6 for resistance distance and Kirchhoff index is shown in Table 5. The maximum error between resistance distance $R$ and $R_{approx}$ is 96.27% for graph G124 for node pair (22, 39).

| Graphs | # Nodes | # Edges | Sequential Time (seconds) | Parallel Time (seconds) |
|---|---|---|---|---|
| graphExample | 25 | 44 | 0.050 | 0.030 |
| G124 | 124 | 318 | 3.98 | 0.340 |
| G250 | 250 | 1283 | 7.352 | 0.598 |
| G500 | 500 | 5120 | 10.01 | 0.852 |

Table 3: Comparison of time for sequential and parallel implementation.

| # Processors | Time (seconds) |
|---|---|
| 4 | 1.472 |
| 8 | 1.035 |
| 16 | 0.824 |
| 30 | 0.598 |

Table 4: Comparison of time for number of processors for graph G250.

| Graphs | # Nodes | Average Error (%) Resistance Distance | Error(%) Kirchhoff Index |
|---|---|---|---|
| graphExample | 24 | 9.8 | 11.8 |
| G124 | 124 | 9.72 | 10.01 |
| G250 | 250 | 9.37 | 9.54 |
| G500 | 500 | 8.58 | 9.13 |

Table 5: Quality performance of three-step approximation algorithm.

The effective resistance between a pair of nodes depends on the number of paths between these nodes and their lengths. So, the main parameters that affect the average percent error for resistance distance in a network are: density of intra-cluster edges within the clusters and density of inter-cluster edges between the clusters, the number of clusters, and the number of steps used in the approximation. The more the number of paths, the less is the resistance distance. Also, the longer the paths, the more is the resistance distance. The average percent error of resistance distance for a network with dense intra-cluster edges and sparse inter-cluster edges is less as compared to the network with dense inter-cluster edges. For a graph of 25 nodes, the error is 11.2% for sparse intra-cluster and inter-cluster edges, 10.8% for dense intra-cluster and sparse inter-cluster edges, and 17.5% for dense intra-cluster and dense inter-cluster edges.

## 5.8 Summary

In this chapter, sequential and parallel algorithms for resistance distance have been proposed. The performance of both the algorithms with respect to execution time have been discussed. In addition, a novel approximation algorithm for resistance distance and Kirchhoff index has been introduced. The parameters of network that affect the approximation algorithm have also been discussed.

# Chapter 6

# Conclusions

## 6.1 Summary

This chapter summarizes the research presented in this dissertation. Chapter 1 provided introduction and appropriate literature review relating to network science. It also discussed the key aspects of network science and types of the networks. The overview of resistance distance and Kirchhoff index were described and the layout of the dissertation structure was also given in this chapter.

Chapter 2 presented an overview of electrical networks along with the topological formulas for network functions. The matrices of graph and their properties were discussed along with the Laplacian spectral graph theory. The Laplacian matrix of a graph plays an important role in the computation of resistance distance and Kirchhoff index. The standard method to obtain resistance distance is via Moore-Penrose pseudoinverse $L^{+}$ of the Laplacian matrix $L$ of a connected graph. To avoid the computational complexity and extraneous efforts of Moore-Penrose pseudo-inverse, a new formula for calculating Kirchhoff index was presented in this chapter. Three proofs of this formula based on the properties of the pseudo-inverse of the Laplacian matrix, topological formula for network functions and basic concepts of electrical circuit theory were presented.

Chapter 3 generalized the notion of Laplacian matrix using the fundamental cutset matrix. The concept of Kirchhoff polynomial of a graph was defined in this chapter. Kirchhoff polynomial expresses Kirchhoff index using the elements of the resistance matrix. In this chapter, two approaches were developed to compute the Kirchhoff index. The first approach is based on a matrix transformation and the second approach uses the concept of Kirchhoff polynomial of a graph.

Chapter 4 provided further advances on the concept of Kirchhoff index. This chapter introduced the concept of Weighted Kirchhoff index of a graph and its relationship to Foster's theorems. Foster's theorem is a very important theorem in the field of electrical network analysis. Foster's theorems can be presented as results involving the sum of weighted resistance distances when the weights are chosen appropriately. This chapter presented a generalization of Foster's theorems that retains the circuit-theoretic flavor and elegance of Foster's theorems. A dual form of this theorem was developed in this chapter.

Chapter 5 proposed sequential and parallel algorithms to compute Kirchhoff index. Kirchhoff index captures the effect of topological structure on the performance of networks. It also captures the path diversity between nodes in a network. Kirchhoff index can be used to determine node betweenness in networks that are of interest in network vulnerability studies. In view of this, an efficient methodology to compute Kirchhoff index is required. A novel three-step approximation algorithm for calculation of resistance distance and Kirchhoff index was introduced in Chapter 5. This chapter discussed the parameters of network that affect the three-step approximation algorithm.

## 6.2 Future Directions of Research

Graphs and networks have been used extensively in many recent applications (e.g., social networks, economy, etc.). For instance, all centrality measures in network are based on the shortest distances between pairs of nodes. Though, in general, all paths must be used in assessing the centrality of a node, shortest paths are used because they are easy to compute.  To mitigate the effect of the approximation of criticality by considering only shortest paths, other measures that capture both the lengths of paths and the number of these paths between nodes need to be investigated (e.g., diffusion distance). Resistance distance and Kirchhoff Index are two such measures.

Resistance distance is a generalization of shortest paths. The shorter a path between two nodes the smaller will be the distance. Also, the more the number of paths, the less will be the distance. Thus, resistance distance captures the impact of both the lengths of paths and the number of paths on criticality measures. On the other hand, Kirchhoff index may be viewed as an aggregate property of a group of nodes (that is, the average of all resistance distances across all pairs of nodes in the group).

We propose two problems for further investigations, employing the notion of resistance and Kirchhoff Index.

### 6.2.1 Graph clustering

In graph clustering one is interested in partitioning the nodes of a graph into non-overlapping clusters satisfying certain additional properties. These additional constraints are defined by the applications considered.  Two extensive reviews of graph clustering that discuss both theoretical advances and some practical heuristics may be found in the reference (Thulasiraman, et al., 2015). The reference (Aluru, 2006) provides a very good coverage of applications of clustering in molecular biology. A general class of clustering algorithms that satisfy the following constraints merits further investigation.

**Determine clusters minimizing the sum of Kirchhoff indices of all clusters subject to a limit on the maximum number of nodes in each cluster.**

Since Kirchhoff Index captures the aggregate value of closeness of nodes in a group, the clusters obtained by the solution of the problem will be the groups of nodes that are very close to each other Additional constraints such as minimizing the number of inter-cluster edges can also be introduced in the above formulation. It is easy to see that this problem is NP-hard. Heuristics such as those based on spectral partitioning, multi-commodity flows etc. for other classes clustering problems are available.

In social network analysis, a related problem called community detection has been studied (Newman, 2010; Easley & Kleinberg, 2010; Malliaros & Vazirgianniz, 2013). In community detection one objective is to get clusters that achieve maximum value of what is called modularity. We would like to add modularity constraint to capture the notion of homophily or assortative mixing in networks. In all these works Laplacian matrix and their eigenvalues play a central role. What makes the clustering problem defined above novel is the use of the notion of Kirchhoff Index. We believe combing with this the idea of assortative mixing will lead to more powerful measures and algorithms for clustering.

### 6.2.2 Similarity and criticality measures

**Similarity:**

Similarity measures based on degree distributions and other topological parameters are available in the literature. We propose a new definition of similarity of nodes using the concept of resistance distance. We first define the Kirchhoff index of a node *v* as the sum of the resistance distances of this node to all other nodes in the network. We define two nodes as similar if their node Kirchhoff index values are equal. Our method to calculate the Kirchhoff index in this dissertation can also be used to compute the node Kirchhoff index values starting from the inverse of the reduced Laplacian.

**Criticality:** Node (edge) betweenness measure used in social network analysis captures the critical value of a node (edge) with respect to the number of paths that pass through the node (edge). For a review of research on this topic, the references (Newman, 2010; Easley & Kleinberg, 2010) may be consulted. We define a new criticality measure based on resistance distance instead of shortest paths. This will allow us to estimate the impact of path lengths and number of paths.

Between measure algorithms, references based on shortest paths are available (Newman, 2010; Easley & Kleinberg, 2010; Brandes, 2001). One new direction of study is to investigate algorithms for betweenness measures using resistance distances instead of shortest paths. It will also be interesting to study the relationship between these new betweenness measures and criticality of nodes and edges with respect to their ability to cause disruption in network functions.

Our focus will be on handling large graphs. We shall also study the impact of a cascade of failures of critical nodes. We have studied problems of this type in different contexts: cascading failures in multi-layer networks and power grids (Zhou, et al., 2012;

Wu, et al., 2017). We plan to build on this expertise to advance knowledge by applying the concept of resistance distances to the study of cancer progression and detection of critical driver mutations.

## Bibliography

Albert, R., Jeong, H. & Barabasi, A. -L., 1999. Diameter of the World Wide Web. *Nature,* Volume 401, pp. 130-131.

Albert, R., Jeong, H. & Barabasi, A. -L., 2000. The Internet's Achilles' heel: Error and attack tolerance of complex networks. *Nature,* Volume 406, pp. 378-382.

Aluru, S., 2006. *Handbook of Computational Molecular Biology.* Chapman & Hall/CRC.

Anon., 2003. *Seven Bridges of Konigsberg.* [Online] Available at: http://en.wikipedia.org./wiki/Seven_Bridges_of_Konigsberg [Accessed 9 November 2016].

Atay, F. M., Biyikoglu, T. & Jost, J., 2006. Synchronization of networks with prescribed degree distributions. *IEEE Trans Circuits Syst. I,* 53(1), pp. 92-98.

Bapat, R. B., 2004. Resistance matrix of a weighted graph. *Commun. Math. Comput. Chem.,* Volume 50, pp. 73-82.

Barahona, M. & Pecora, L. M., 2002. Synchronization in small-world systems. *Phys. Rev. Lett.* , Volume 89, p. 054101.

Barbasi, A. -L., Albert, R. & Jeong, H., 1999. Emergence of scaling in random networks. *Science,* Volume 286, pp. 506-512.

Bonacich, P., 1972. Factoring and weighing approaches to clique identification. *J. Math. Sociol.,* Volume 2, pp. 113-120.

Bonchev, D., Balaban, A. T., Liu, X. & Klein, D. J., 1994. Molecular cyclicity and centricity of polycyclic graphs. I. Cyclicity based on resistance distance or reciprocal distances. *Int. J. Quan. Chem.* , Volume 50, pp. 1-20.

Brandes, U., 2001. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology,* 25(2), pp. 163-177.

Cinkir, Z., 2007. *The tau constant of metrized graphs.*Thesis at the University of Georgia.

Cinkir, Z., 2011. Generalized Foster's identities. *International Journal of Quantum Chemistry,* 111(10), pp. 2228-2233.

Coppersmith, D., Doyle, P., Raghavan, P. & Snir, M., 1990. Random walks on a weighted graphs and applications to online algorithms. *In Proceedings of the 22nd symposium on the Theory of Computing* , pp. 369-378.

Corporation, N., 2010. *NVIDIA CUDA C Programming Guide,* version 3.2.

Dorogovtsev, S. N. & Mendes, J. F. F., 2002. Evolution of networks. *Adv. Phys.,* 51(4), pp. 1079-1187.

Dorogovtsev, S. N. & Mendes, J. F. F., 2003. *Evolution of networks - from biological nets to internet and WWW.* Oxford Univ. Press.

Dorogovtsev, S. N., Mendes, J. F. F. & Samukhin, A. N., 2000. Structure of growing networks with preferential linking. *Phys. Rev. Lett.,* 85(21), pp. 4633-4636.

Dorogovtsev, S. N., V., G. A. & Mendes, J. F., 2002. Pseudofractal scale-free web. *Phys. Rev. E,* 65(6), p. 066122.

Doyle, P. G. & Snell, J. L., 1984. Random walks and electric networks. *Carus Mathematical Monographs,* Volume 22.

Easley, D. & Kleinberg, J., 2010. *Networks, Crowds and Markets.* New York: Cambridge University Press.

Erdos, P. & Renyi, A., 1960. On the evaluatio of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.,* Volume 5, pp. 17-61.

Euler, L., 1736. Solutio problematis ad geometriam situs pertinetis. *Comment. Acad. Sci. U. Petrop.,* Volume 8, pp. 128-140.

Faloutsos, M., Faloutsos, P. & Faloutsos, C., 1999. *On power-law relationships of the Internet topology.* New York, Proc. ACM SIGCOMM'99 conf. ACM Press, pp. 251-262.

Foster, R. M., 1949. The average impedance of an electrical network. *Contributions to Applied Mechanics, (Reissner Anniversary Volume), Ed. J. W. Edwards,* pp. 333-340.

Foster, R. M., 1961. An extension of a network theorem. *IRE Transaction on Circuit Theory,* Volume 8, pp. 75-76.

Gilbert, E. N., 1959. Random graphs. *Ann. Math. Stat.,* 30(4), pp. 1141-1144.

Granovetter, M. S., 1973. The strength of weak ties. *Am. J. Sociol,* 78(6), pp. 1360-1380.

Gutman, I. & Mohar, B., 1996. The Quasi-Wiener and the Kirchhoff indices coincide.

*J. of Chem. Inf. Comput. Sci.,* Volume 36, pp. 982-985.

Gutman, I. & Xiao, W., 2004. Generalized inverse of the Laplacian matrix and some applications. *Bulletin de Academie Serbe desScience at des Arts (Cl. Math. Natur.),* Volume 129, pp. 15-23.

Holland, J. H., 1998. *Emergence: From Chaos to Order.* Reading, MA: Addison-Wesley.

Hong, H. C. M. Y. & Kim, B. J., 2002. Synchronization on small-world networks. *Phys. Rev. E,* Volume 65, p. 026139.

Hu, Z., Thulasiraman, K. & Verma, P., 2013. Complex Networks: Traffic Dynamics, network perofrmance and network structure. *Americal Journal of Operations Research,* 3(1A), pp. 187-195.

Hu, Z., Verma, P. K. & Thulasiraman, K., 2013a. *Interplay between traffic dynamics and network structure.* Seville, Spain, IARIA 8th International Conf. on Systems (ICONS), pp. 202-206.

Johnson, D. S., Aragon, C. R., McGeoch, L. A. & Schevon, C., 1989. Optimization by simulated annealing: an experimental evaluation. Part I, graph partitioning. *Oper. Res., INFORM, Institute for Operations Research and the Management Sciences (INFORMS),* Volume 37, pp. 865-892.

Jost, J. & Joy, M. P., 2002. Spectral properties and synchronization in coupled map lattices. *Phys. Rev. E,* 65(1), p. 016201.

Karypis, G. & Kumar, V., 2013. *MeTis: A software package for partitioning unstructured graphs, partitoning meshes, and computing fill-reducing orderings of sparse matrices,* Univ. of Minnesota, Minneapolis, MN: Version 5.1.0, http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/manual.pdf.

Kernighan, B. W. & Lin, S., 1970. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal,* pp. 291-307.

Kirchhoff, G., 1847. Uber die Auflosung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Strome gefiihrt wird. *Ann. Phys. Chem.,* Volume 72, pp. 497-508.

Kleinberg, J. M., 2000. Navigation in a small world. *Nature,* Volume 406, pp. 845.

Klein, D. J., Lukovits, I. & Gutman, I., 1995. On the definition of the Hyper-Wiener Index for cycle-containing structures. *J. Chem. Inf. Comput.,* Volume 35, pp. 50-52.

Klein, D. J. & Randic, M., 1993. Resistance distance. *J. Math. Chem.,* Volume 12, pp. 81-95.

Kunz, M., 1993. On topological and geometrical matrices. *J. Math. Chem,* Volume 13, pp. 145-151.

Kuramoto, Y., 1984. *Chemical Oscillations, Waves, Turbulence.* Springer-Verlag.

Lewis, T. G., 2009. *Network Science: Theory and Application.* Wiley.

Liu, J., 2003. *General complex dynamical network models and its synchronizaton criterions.* Yichang, China, Proc. 22nd Chinese Control Conf., pp. 380-384.

Liu, J., Yu, X. & Chen, G., 2004. Chaos synchronization of general complex dynamical networks. *Physica A,* Volume 334, pp. 281-302.

Liu, J., Yu, X., Chen, G. & Cheng, D., 2004a. Characterizing the synchronizability of small world dynamical networks. *IEEE Trans. Circuits Syst. I,* Volume 51.

Liu, J., Zhou, T. & Zhang, S., 2002. Chaos synchronization between linearly coupled chaotic system. *Chaos Solut. Fractal,* Volume 144, pp. 529-541.

Lukovits, I., Nikolic, S. & Trinajstic, N., 1999. Resistance distance in regular graphs. *Int. J. Quan. Chem.,* Volume 71, pp. 217-225.

Malliaros, F. D. & Vazirgianniz, M., 2013. Clustering and Community Detection in Directed Networks: a Survey. *Physics Reports,* 533(4), pp. 95-142.

Molitierno, J. J., 2012. *Applications of combinatorial matrix theory to Laplacian matrices of graphs.* Chapman and Hall/CRC.

Newman, M., 2010. *Networks: An Introduction.* New York: Oxford University Press Inc..

Palacios, J. L., 2001. Closed-form formulas for Kirchhoff index. *Int. J. Quantum Chem,* Volume 81, pp. 135-140.

Palacios, J. L., 2004. Foster's formula via probability and the Kirchhoff index. *Methodology and Computing in Applied Probability,* Volume 6, pp. 381-387.

Swamy, M. N. & Thulasiraman, K., 1981. *Graphs, Networks and Algorithms.* Wiley-Inter-science.

Tetali, P., 1991. Random walks and the effective resistance of networks. *J. Theoret. Probab,* 4(1), pp. 101-109.

Tetali, P., 1994. An extension of Foster's network theorem. *Combin. Probab. Comput.,* Volume 3(3), pp. 421-427.

Thulasiraman, K. (Editor in Chief), Arumugam, S., Brandstaedt, A. & Nishizeki, T. (Editors), 2015. *Handbook of Graph Theory, Combinatorial Optimization and Algorithms.* CRC Press.

Thulasiraman, K., Jayakumar, R. & Swamy, M. N., 1983. Graph theoretic proof of a network theorem and some consequences. *Proc. IEEE* , Volume 71, pp. 771-772.

Thulasiraman, K. & Swamy, M. N., 1992. *Graphs: Theory and Algorithms.* New York: John Wiley & Sons.

Wang, X. & Chen, G., 2002a. Synchronization in small-world dynamical networks. *J. Bifurcation Chas,* 12(1), pp. 187-192.

Wang, X. & Chen, G., 2002b. Synchronization in scale-free dynamical networks: Robustness and fragility. *IEEE Trans. Circuits Syst. I,* Volume 49, pp. 54-62.

Wang, X. & Chen, G., 2002. Pinning control of a scale-free dynamical networks. *Physica A,* Volume 310, pp. 521-531.

Wang, Z., Chakrabarti, D., Wang, C. & Faloutsos, C., 2003. *Epidemic spreading in real networks: an eigenvalue viewpoint.*, Proc. 22nd Intnatl. Symp. Reliable Distributed Systems.

Watts, D., 1999a. *Samll Worlds.* Princeton, NJ: Princeton Univ. Press.

Watts, D. J., 1999. Networks, dynamics, and the small-world phenomenon. *Am. J. Sociol.,* 105(2), pp. 493-527.

Watts, D. J. & Strogatz, S. H., 1998. Collective dynamics of "small world" networks. *Nature,* Volume 393, pp. 440-442.

Waxman, B. M., 1988. Routing of multipoint connections. *IEEE J. Select. Areas Commun.,* 6(9), pp. 1617-1622.

Wu, D. et al., 2017. A study of the impacts of flow direction and electrical constraints on vulnerability assessment of power grid using electrical betweenness measures. *Physica A: Statistical Mechanics and its Applications,* Volume 466, pp. 295-309.

Xiao, W. & Gutman, I., 2003a. Resistance distance and Laplacian spectrum. *Theor. Chem. Acc.,* Volume 110, pp. 284-289.

Xiao, W. & Gutman, I., 2003. On resistance matrices. *MATCH Commun. Math. Comput. Chem.*, Volume 49, pp. 67-81.

Yule, G. U., 1925. A mathemetical theory of evolution based on the conclusions of Dr. J. C. Willis. *Phil. Trans. Roy. Soc. Lond. B,* Volume 213, pp. 21-87.

Zhang, H. & Yang, Y., 2007. Resistance distance and Kirchhoff index in circulant graphs. *Int. J. Quantum Chem,* Volume 107, pp. 330-339.

Zhou, B. & Trinajstic, N., 2009. On resistance-distance and the Kirchhoff index. *J. Math. Chem,* Volume 46, pp. 283-289.

Zhou, Z. et al., 2012. *Novel Survivable Logical Topology Routing in IP-over-WDM Networks by Logical Protecting Spanning Tree Set.* St. Petersburg, Russia, Proc. IEEE Reliable Network Design and Modeling Conference.

Zhu, H. Y., Klein, D. J. & Lukovits, I., 1996. Extensions of the Wiener Number. *J. Chem. Inf. Comput. Sci.,* Volume 36, pp. 420-428.