UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DESIGN OF TRANSIENT ENERGY AUTOMATON AND STUDY OF ENERGY FLOWS ON LATTICE GRAPH

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

XIN LI

Norman, Oklahoma
2018

DESIGN OF TRANSIENT ENERGY AUTOMATON AND STUDY OF ENERGY FLOWS ON LATTICE GRAPH

A DISSERTATION APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

DEDICATION

to

My family

For

Encouraging me to follow my dreams

Acknowledgments

The five years of the Ph.D. program at University of Oklahoma has been an incredible journey in my life, and I would like to express my gratitude to many people for their help and supports during this journey.

Firstly, I would like to thank my advisor Prof. Jiang for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this dissertation. I could not have imagined having a better advisor and mentor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Runolfsson Thordur, Prof. Kash A. Barker, Prof. Ronald D. Barnes, Prof. Thulasiraman Krishnaiya, Prof. Paul Moses and Prof. Choon Yik Tang, for their insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

I would like to thank my family: my parents, Weihua Li, YanHua Geng, and my girlfriend, Zhilin Zhang, for supporting me spiritually throughout writing this dissertation and my life in general. These encouragements from my family have always motivated me to be passionate and confident in my whole research life. And Jian Zeng, hope we can cooperate with each other in the nearly future.

I thank my labmates, Dr. Lihua Zhao, Dr. Malyscheff Alexander, Wanghao Fei, Guomin Ji, Sharma Dhruv, Lei Jiang, Varun Perumalla and Chenxi Lin, for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last five years.

My life at OU has been really wonderful and memorable, special thank to Dr. Cynthia Rogers, Dr. Stephen Ellis, Dr. Pat Meirick , Dr. Lee Stitzel, Weishan Liang and Dr. Wayne Riggs, for the greatest football and English culture class ever in Oklahoma every Saturday morning.

And finally, last but by no means least, thank my friend at Samsung Research America, Jilong Kuang, Benwen Zhang, Hujun Cui, Leonardo Jimenez, Yolanda Correia, Philip Park, Yoshiya Hirase, Jiang Gao etc. I am quite satisfied with working with you and appreciate for all of the suggestions.

Thanks for all your encouragement!

Contents

List of Tables

List of Figures

Abstract

The effort of this doctoral dissertation research is devoted to the design of an automaton system and corresponding computer experiments for studying the dynamics of energy flow on lattice graphs. Such a Transient Energy Automaton (TEA) is a computational system that automates dynamic energy exchanges on a graph once triggered by disturbances similar to those used in the analysis of the dynamics of multi-machine power model system. The differences between TEA and other automata are: 1) TEA considers all of the vertices in the lattice graphs that are connected with each other; 2) TEA supposes the energy exchange processing by two specific actions; 3) TEA defines the governing rules of those actions.

The TEA enables two fundamental actions of energy exchange at each vertex: a slow action to absorb a certain fraction of the energy received from the site of initial disturbance or the neighboring vertices, and a fast action to reflect the rest of the received energy to the rotors or oscillators at neighboring vertices on the graph. These two fundamental actions coupled together to enable the energy flow of TEA.

In order to simplify the coupling of the two actions in the TEA, we let these two fundamental actions at local site or vertex be governed by the following rules:

- For each vertex, the amount of energy received at a local site in the reflection action, which is defined as dispersion ratio, is a function of the graph geometry including the structure of lattice and value configuration of edges;

- The fraction of energy that can be absorbed by each vertex in the absorption action, which is defined as absorption ratio, is based on the capability of the

vertex;

- We suppose at a particular time instant, all the vertices are in one same action, either fast action or slow action;
- Such rules are functions of the geometry and value configuration of graph and the property of vertices, which ultimately enable the TEA to create time evolution of energy distribution, that may affect the dynamics of the system.

The two governing rules apply to the fast and slow actions at the local site or vertex-level by the ratios of dispersion and absorption. The ratios are defined by two major intrinsic factors for the global energy exchange: 1) geometry of graph and its value configuration, and 2) property of vertex. The geometry controls the energy dispersion among vertices according to their propinquities and centralities configured by graph lattice, while the property of vertex is the capability to absorb or reflect the energy received.

The energy distribution among vertices on a graph and its time evolution are computed recursively using computational formula of the two actions under the governing rules. Moreover, the computation of the accumulated energy at each vertex satisfies an additive superposition principle to obtain the homogeneity of energy distribution on graph for each time step and for the time evolution of the process. The global properties of energy flow on a graph can then be investigated by analyzing the functionalities of graph geometry and vertex property at local site or vertex-level, both directly impact the fast and slow actions.

The experiment with TEA concentrates on discovering the linkage between characteristics of graph, such as homogeneously and heterogeneously distributed, and the dynamics of TEA from the local-level or global-level. More specifically, the proposed TEA model system and the experiment approach can simulate energy exchange at each local site or vertex level to obtain the global dynamics of energy flow on graph.

Through analyzing energy absorption at vertices in slow actions, energy dispersion over the lattice in fast actions, homogeneity of the energy distribution on graphs, as well as their time evolution, we are able to obtain constructive evidence that energy automation may possibly be a simplifying approach to understanding the global dynamical properties of energy flow on graphs, and more ambitiously, those found in real large-scale energy interconnections.

The automaton model system and the experiment approach exhibits a number of advantages: practically they have a simplifying complex feature in modeling coupled dynamics on graph to produce global features of large-scale energy interconnections, demanding an extra-low computational burden comparing to those needed to solve Differential Algebraic Equations of large-scale system. More importantly, a quick insight on the nature of dynamics on graph that ordinary methods are inadequate, particularly when there are significant structural changes in graph topology and vertex functionality.

Chapter 1

Introduction

Energy analysis is one of the feasible and practical methods for gaining insight into a system. Based on the time dependency, energy analysis can be categorized as static energy analysis and dynamic energy analysis [1] [2]. The static energy analysis evaluates the instantaneous energy distribution configuration of the system [3] [4]. This method is not focused on tracking short-run time-series evolution of energy flow in a system. Rather than static energy analysis, dynamic energy analysis takes the interactive nature of energy into account, and tracks the energy injected into every element of the system and the energy dissipation out of them [5]. Thus, dynamic energy analysis is commonly used by the engineers. For example, Energy2D is a simulation program that models dynamics of heat-transfer process [6]. These dynamic energy analysis approach usually based on solving dynamic equations and which is very time-consuming.

Cellular Automata (CAs) are simple mathematical idealizations of nature systems and these are designed to follow predetermined set of rules [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21]. The classical model of CAs is to define a set of local rules and use these local rules updating the state of each cell and its immediate surrounding neighbor cells step by step. Even with a set of simple local rules, CAs could produce a quite complicated behavior. We will briefly introduce CAs in the Chapter 2. CAs are used in the research of several different areas, for example, [14] use CAs to simulate the traffic flow.

The objective of this dissertation is to design an automaton system, which we named as Transient Energy Automaton (TEA), and to use the corresponding computer experiments for analyzing the dynamics of energy on the graph. When comparing the TEA with the CAs, we can draw the following conclusions:

- Both of these models are complex systems generated from a simple configuration and a simple set of local rules;
- Both of these two models are discrete models build upon a collection of vertices (or called cells in CAs) on a graph that evolve through discrete time;
- Unlike CAs, where all the cells could be in different states at a time, TEA model assumes that all vertex in the graph must be in a same state, which is in either fast action or slow action, at a time;
- Unlike CAs, where each cell tends to interact locally, each vertex in TEA model interacts globally with all of the other vertices on the graph;
- Unlike CAs, where the next status of each cell is depending on current states of this cell and its corresponding neighborhood, the energy contained of each vertex in the TEA model at next time step is depending on the energy distribution on the graph at current time;
- Unlike CAs where the configuration is the states distribution on the graph, the configuration of TEA is the energy distribution on the graph at each step;
- Both of those two models reflect the energy spreads from local area to the global area;

we will evaluate TEA model by comparing the TEA predicted results to the ones predicted based on the power system dynamic simulations in a power grid.

The dissertation is organized as follows: In Chapter 2 we overview the important concepts and methods that forms the basis for the proposed research. In Chapter 3 we introduce the inspiration of this research. In Chapter 4 we show the model of Transient Energy Automaton. In Chapter 5 we present the experimental system. In Chapter 6 we discuss impacting factors of energy distribution dynamics in TEA system. Chapter 7 we show comparative study. In Chapter 8 we conclude the research presented in this dissertation, and recommendations for future research.

Chapter 2

Background

Transient Energy Automaton (TEA) is an automaton for analyzing the dynamics of energy on the graph. It is an interdisciplinary area, where different aspects of graph theory, formal language theory, automaton theory, and data visualization must be considered all together. Therefore, in this chapter we will introduce the related concepts and methods necessary to design TEA.

This chapter consists of two sections: section 2.1 briefly introduces cellular automata; and section 2.2 introduces relevant approaches in graph layout.

2.1 Automata Modeling in Computer Science

Cellular Automata (CAs) are discrete dynamical systems that use simple, local rules to simulate complex behavior of the graph. Before introducing important concepts in CAs, we will briefly describe related concepts in general automata theory.

2.1.1 Related Terminologies of General Automata

Set: A set is a collection of unique elements. For example, the set S of symbols x, y, z is denoted as $S = \{x, y, z\}$. In order to indicate that x is a member of the set

S, we write $x \in S$; otherwise, we write $x \notin S$. A set S_1 is said to be a subset of set S if every element of S_1 is also an element of S, denoted by $S_1 \subseteq S$. A set S_1 is said to be a proper subset of S if $S_1 \subseteq S$. If a set S contains elements not in S_1, we denote it by $S_1 \subset S$. A set is said to be a finite set if it contains a finite number of elements; otherwise it is called an infinite set. The usual set operations are union ($\cup$), intersection ($\cap$), and difference(-), which are defined as:

- $S_1 \cup S_2 = \{x : x \in S_1 \; or \; x \in S_2\}$,
- $S_1 \cap S_2 = \{x : x \in S_1 \; and \; x \in S_2\}$,
- $S_1 - S_2 = \{x : x \in S_1 \; and \; x \notin S_2\}$,

Function: A function is the relation between a set of inputs and a set of permissible outputs with the property that each input is related to exactly one output. If f denotes a function, then the input set is called the domain of f while the output set is the range of f. We write $f : S_1 \to S_2$ to indicate the domain of f is a subset of S_1 while the range of f is a subset of S_2.

Alphabet: An alphabet is a finite and non-empty set of symbols, such as the binary alphabet $\Sigma = \{0, 1\}$. The alphabet of an automaton is usually denoted as Σ.

Word (or string): A word is a finite sequence of symbols selected from a alphabet. For example, "01010"is a word from the binary alphabet $\Sigma = \{0, 1\}$. The set of all words over an alphabet Σ is denoted by Σ^*. We denote the empty string as ϵ. For example, if $\Sigma = \{0, 1\}$, then $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, ...\}$.

Languages: The languages are a collection of words of finite length, which are all constructed from a finite alphabet of symbols. For example, let $\Sigma = \{0, 1\}$, then $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, ...\}$; the set $\{0, 11, 110\}$ is a language on Σ.

Grammar: The Grammar is a finite collection of rules that can describe an infinite language. A grammar G is defined as a quadruple $G = (N, \Sigma, P, S)$ where

- N is a finite set of nonterminals,

- Σ is a finite set of terminals,
- $S \in N$ is the start symbol,
- P is a finite subset of $N \times V^*$, which are called the set of production rules. Here, $V = N \cup \Sigma$,

Context-Free Grammar: A given grammar $G = (N, \Sigma, P, S)$ is a context-free grammar if all productions in P have the form $A \to x$, where $A \in N$ and $x \in (N \cup \Sigma)^*$.

Automaton: Automaton is an abstract machine which is designed to perform a function according to a given sequence of inputs in discrete time steps. The input file of an automaton is a word over a given alphabet, which is read only for automaton. The input mechanism for automaton is read from left to right, one symbol at a time; also, automaton can detect the end of each input word. Each automaton contains a finite number of states. The purpose of these states is to remember the relevant portion of the system's history [22]. We suppose the automaton can only be in one of a given state at a given time. The state of the automaton at the next time step is determined by transition function. This transition function identifies the next state by recorded states and current input symbol. There are many applications of automaton, such as, software for scanning large bodies of text for finding pattern, etc.

Finite Automaton: Finite Automaton is a simple idealized machine used to recognize patterns of the input. This type of Automaton has no temporary storage to record the previous status. A finite automaton M can be defined by a quintuple $M = \{Q, \Sigma, \delta, q_0, F\}$, where

- Q: is a finite set of states;
- Σ: is input alphabet;
- $q_0 \in Q$: is start state;

- $F \subseteq Q$: is a set of accept states;
- $\delta : Q \times \Sigma \to Q$: is a transition function;

Pushdown Automaton: Pushdown Automaton provides a way to implement a context-free grammar. Compared with Finite Automaton, this type of automaton can memorize an infinite amount of information by implementing a stack into its structure. A Pushdown Automaton M can be defined by a sextuple $M = \{Q, \Sigma, \Gamma, \delta, q_0, z, F\}$, where

- Q: is a finite set of states;
- Σ: is input alphabet;
- Γ: is stack alphabet;
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to$ set of finite subsets of $Q \times \Gamma^*$: is a transition function;
- $q_0 \in Q$: is start state;
- $z \in \Gamma$: is the initial stack top symbol;
- $F \subseteq Q$: is a set of accept states;

Turing Machine: Turing Machine is a type of automaton whose temporary storage is a tape. This tape is divided into several cells, each of those cells is capable of holding one symbol. The Turing machine can travel right or left on the tape while read and write a single symbol on each move. A Turing Machine M can be defined by a sextuple $M = \{Q, \Sigma, \Gamma, \delta, q_0, b, F\}$, where

- Q is a finite set of states;
- Σ is input alphabet;
- Γ is tape alphabet;

- δ: is a transition function;
- $q_0 \in Q$: is start state;
- $b \in \Gamma$: is the blank symbol;
- $F \subseteq Q$: is a set of accept states;

2.1.2 Description of Cellular Automata

Structure: A CA consists of elements which we called cells; each cell can be in one of the finite states set at a time. The graph of cells can be in any finite number of dimensions. To simplify, in this section, we discuss CA in one-dimension and the state for each cell is either in white or black.

Neighborhood: The neighborhood of a cell is the set of adjacent cells. In one-dimensional CA, the neighborhood is described by the radius r, which refers the number of cell, either left or right, from the current cell. In the following introduction, we assume that the radius r is defaulted by one.

Local interactions: Each cell's behavior depends only on what happens within its local neighborhood of cells.

Evolution: The evolution of CA depends on the cells' states changing pattern via time, which is determined by CA rule. When the cells change from current states to the next states, each cell looks around and gathers neighborhood states' information. Based on the cell's current state, its neighbors' current states, the rule will determine the next state of the cell. In CA, we suppose all of the cells change their states at the same time.

Configuration: Configuration is the concatenation of all cells states on the graph of CA at time t.

Definition: A d-dimensional CA is a quadruple $A = (Z^d, S, N, \delta)$ where:

- Z^d: is the discrete graph where each element of this lattice is the cell of the CA;

- S: is a finite set of states;
- N: stands for the neighborhood of each cell in A;
- δ: is the rule of CA.

Fractal: Fractal is generated by the evolution of CA, and which reflects the configuration changes via time. In this chapter, we suppose the fractal generated by CA starting from only the middle cell is colored as black. After initiation, the cells in all following rows are colored automatically depending on the rule and related neighborhood.

Classes of fractal pattern: By different rules, CAs can produce varieties of fractal patterns. These fractal patterns can be characterized as four classes, including:

- Class l: CA evolution leads to a homogeneous state, where all cells stably end up with the same value;
- Class 2: CA evolution leads to a set of stable or simple periodic structures;
- Class 3: CA evolution leads to a chaotic, non-periodic pattern;
- Class 4: CA evolution leads to complex pattern and structure.

Figure 2.1 shows the examples of different classes of fractal pattern generated by CAs.

We can summarize the features of CAs as follows:

- CA is a complex system, generated from a very simple configuration and simple rules;
- CA is a discrete model built upon a collection of cells on a graph that evolves through a number of discrete time;
- Each cell of CA is found in one of a finite number of states;

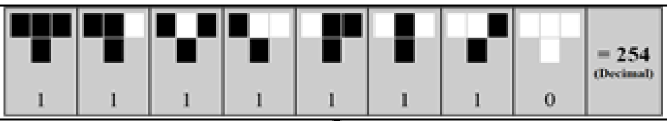
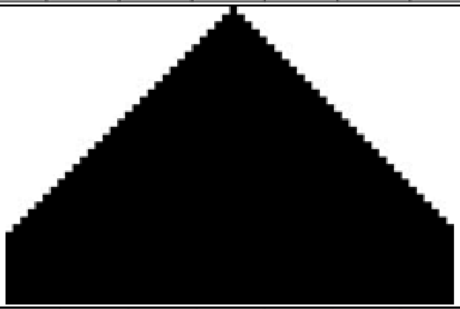
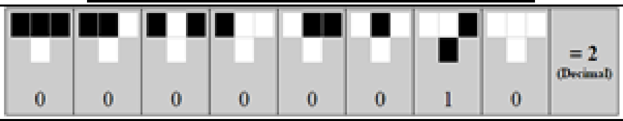
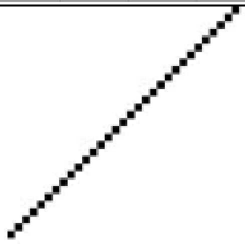
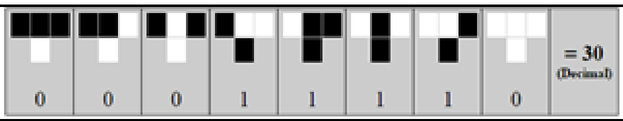
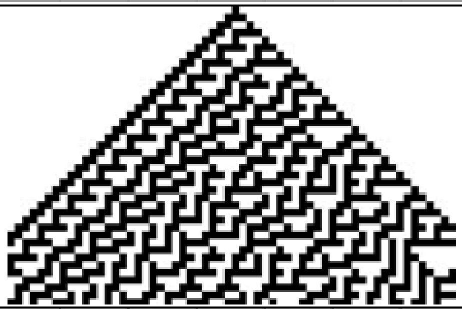
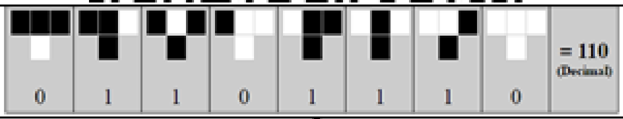
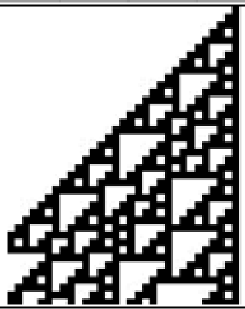
Class 1	Rules(254)	
	Fractal	
Class 2	Rules(2)	
	Fractal	
Class 3	Rules(30)	
	Fractal	
Class 4	Rules(110)	
	Fractal	

Figure 2.1: Examples of different classes of fractal pattern generated by CAs.

- Each cell of CA tends to interact locally depending on itself and its neighborhood;
- The state of each cell in CA at the next time is determined by the current state of the cell and its neighborhood;
- The configuration of CA is the concatenation of all cells' states on the graph;
- The pattern generated by CA can be seen as the energy spreads from local area to the global area.

2.2 Graph Layout

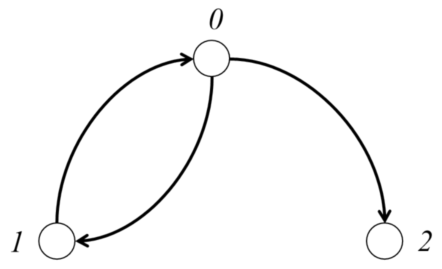
In order to help people understand the topology structure of the TEA graph, we deploy the graph drawing approach in this dissertation. Our approach aims at exhibiting the graph structure onto a two-dimensional display automatically and revealing information buried inside. This technique is a combination of Mathematics and Computer Science, which comes from subjects of graph theory, topology, and information visualization etc.

2.2.1 Related Terminologies of Graph Theory

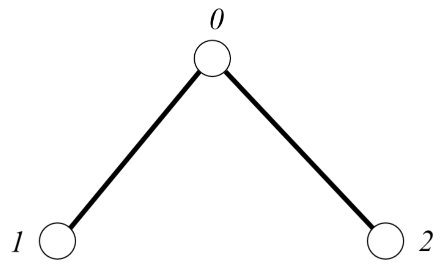
Graph: A graph G can be defined as $G = (V, E)$, where $V = \{v_0, v_1, ..., v_{|V|}\}$ stands for a set of vertices (also called as nodes or points) and $E = \{e_0, e_1, ..., e_{|E|}\}$ stands for a set of connections (also called as edges or lines or arcs). The order of a graph is $|V|$, which stands for the number of vertices in the G; the size of a graph is $|E|$, which stands for the number of connections in the graph G.

Directed graph: Given a graph $G = (V, E)$, if every connection in this graph acts as an ordered pair, we call this graph is directed graph. Figure 2.2a shows an example of directed graph.

Undirected graph: Given a graph $G = (V, E)$, if every connection in this graph acts as an unordered pair, we call this graph is undirected graph. Figure 2.2b shows an example of undirected graph.



(a) An example of directed graph.



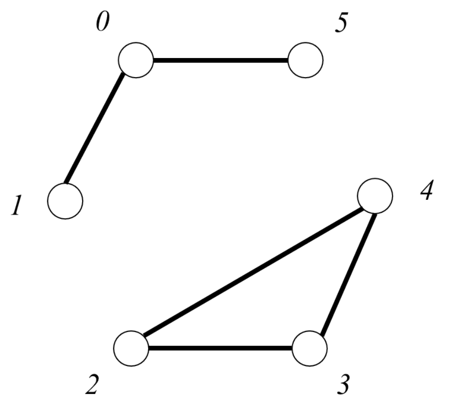
(b) An example of undirected graph.

Figure 2.2: Directed graph vs undirected graph.

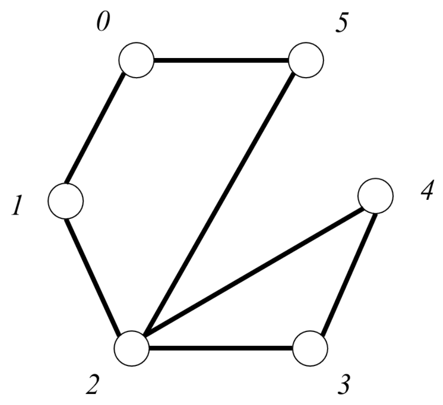
Disconnected graph: Given an undirected graph $G = (V, E)$, if there exists two vertices with no path between them, we call this graph is disconnected graph. Figure 2.3a shows an example of disconnected graph.

Connected graph: Given an undirected graph $G = (V, E)$, if for every pair of vertices there exists at least one path between them, we call this graph is connected graph. In a connected graph, there are no unreachable pair of vertices. Figure 2.3b shows an example of connected graph.

Complete graph: Given a connected graph $G = (V, E)$, if every pair of vertices in this graph is connected by an connection, we call this graph the complete graph. A complete graph contains all possible connections. Figure 2.4 shows an example of complete graph.



(a) An example of disconnected graph.



(b) An example of connected graph.

Figure 2.3: Disconnected graph vs connected graph.

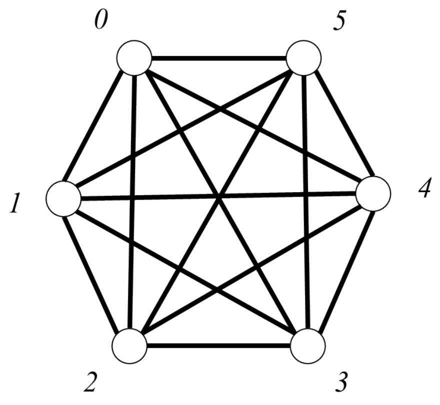


Figure 2.4: An example of complete graph.

Weighted graph: In many applications, there is a numerical value associated with each connection of a graph. We define this value is the weight of a connection and remark this kind of graph the weighted graph. The weight could be a measure of the length of a route, the capacity of a line, the energy required to move between locations along a route, etc. In this dissertation, we assume the weight represents a measure of the length between each pair of vertices.

Shortest path between vertices in a graph: Given a weighted graph and a designated pair of vertices v_i and v_j, we could find a path of least total weight from v_i to vertex v_j in the graph. We call that path the shortest path between vertex v_i and vertex v_j. Figure 2.5 exhibits an example of weighted graph, which is also a complete graph. The shortest distance between vertex 3 and vertex 2 is 6. For a weighted graph G, there may be more than one shortest path between two vertices. If there is no path connecting the two vertices, then the shortest distance between these two vertices is defined as infinite.

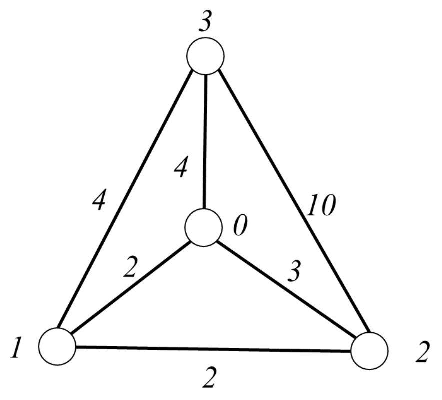


Figure 2.5: An example of weighted graph.

Degree: Degree is the number of connections to the vertex. The degree of a vertex v is denoted as $\deg(v)$. For example, the degree of vertex 0 in Figure 2.5 is 3.

Degree Matrix: Given a graph $G = (V, E)$ with $|V| = n$, the degree matrix D for G is a $n \times n$ diagonal matrix defined as:

$$D_{ij} = \begin{cases} \deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

For example, the degree matrix of graph in Figure 2.5 is:

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix} \tag{2.2}$$

Adjacency Matrix: Given a graph $G = (V, E)$ with $|V| = n$, the degree matrix A for G is a $n \times n$ matrix defined as:

$$A_{ij} = \begin{cases} 1 & \text{if vertex } i \text{ and vertex } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

For example, the adjacency matrix of graph in Figure 2.5 is:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{2.4}$$

Laplacian Matrix: Given a graph $G = (V, E)$, the laplacian matrix L for G is $L = D - A$. For example, the Laplacian matrix of graph in Figure 2.5 is:

$$L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix} \tag{2.5}$$

Planar: A planar graph is a graph that can be drawn on the plane without any connections intersect with each other out of their endpoints.

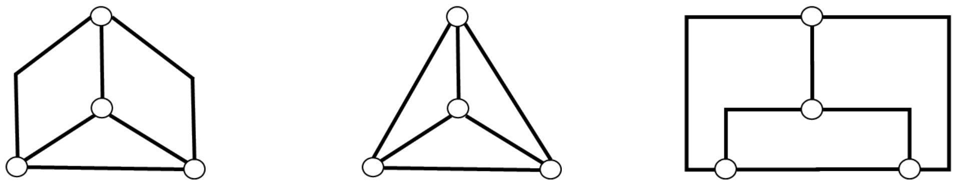
Graph Energy: Given a graph $G = (V, E)$. Let A be the adjacency matrix of G and let λ_i where $i = 1, ..., |V|$ be the eigenvalues of A. Then graph energy is defined as the sum of the absolute values of the eigenvalues.

$$E(G) = \sum_{i=1}^{n} |\lambda_i|$$

.

Drawing Conventions: They are approaches of how to place connections on the layout when creating the final drawing. There are three kinds of drawing conventions, such as:

- Polyline drawing: the final drawing can have any number of bends on a connection. Sub figure 2.6a shows an example of polyline drawing.
- Straight-line drawing: the final drawing has only straight lines in it. Sub figure 2.6b shows an example of straight-line drawing.
- Orthogonal drawing: the final drawing has only vertical or horizontal line segments in it. Sub figure 2.6b shows an example of orthogonal drawing.



(a) An example of polyline drawing. (b) An example of straight-line drawing. (c) An example of orthogonal drawing.

Figure 2.6: Examples of different drawing conventions.

Aesthetic rules: Aesthetic rules are defined for determining the ability of drawing on conveying information [23] [24], such as:

- Minimal edge crossing;
- Vertices and connections must be evenly distributed;
- Curves must be straight lines and should all have the same length;
- Maximal angular resolution;
- Maximize display of symmetries;
- Minimal the drawing area;

Trade-offs: One drawing cannot simultaneously satisfy all of the aesthetic rules, thus some aesthetic rules conflict with one another. For example, in some cases it would be best to include bends in order to avoid edge crossing. Since that, trade-offs need to be made based on the drawing requirement of aesthetic rules.

Drawing Constraints: Some readability aspects require knowledge about the semantics of the specific graph. Since that, constraints are provided as additional inputs to a graph drawing algorithm. For example, in the thesis [25], we use the voltage of each vertices as a constraint to draw the electrical power system.

2.2.2 Description of Graph Layout

There are many kinds of graph layout algorithms [24], such as the force-directed layout algorithm, isometric layout algorithm, random layout algorithm, and circular layout algorithm, etc.. Figure 2.7 shows four different drawings of the same abstract graph.

Isometric layout algorithm: It is a method for the visual representation of three-dimensional nodes in two dimensions. This algorithm uses an isometric perspective to visualize networks. Subfigure 2.7a shows an example of graph generated by isometric layout algorithm.

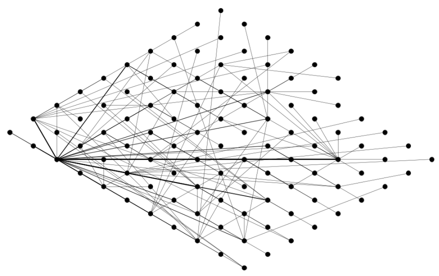
Circular layout algorithm: It is a visualization approach of a graph with the following characteristics:

- The vertices of graph are placed onto the circumference of an embedding circle;
- Each connection of the graph is drawn as a straight line.

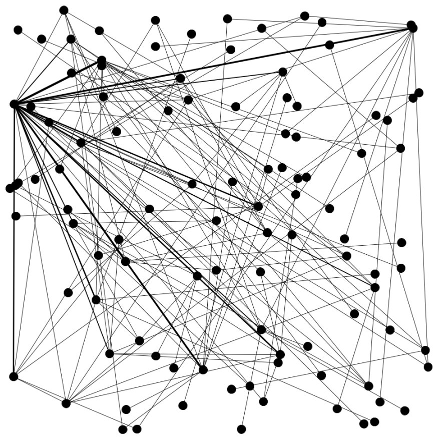
Subfigure 2.7c shows an example of graph generated by circular layout algorithm.

Random layout algorithm: It is a visualization approach of a graph with the following characteristics:

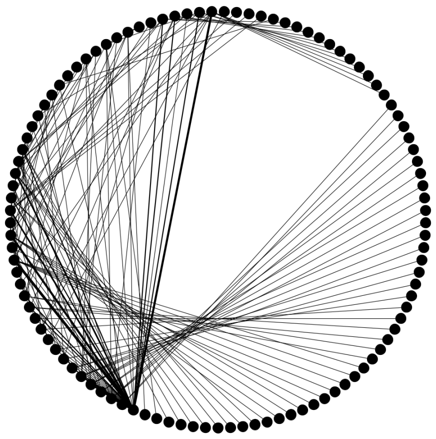
- The vertices of graph are placed randomly onto a two-dimensional display;



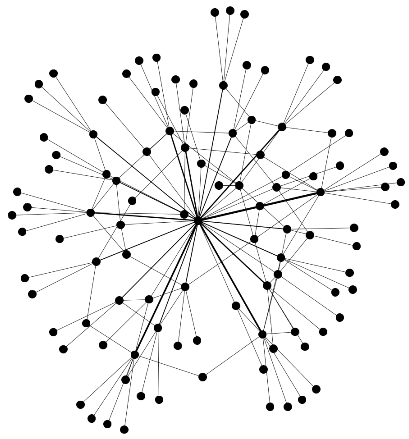
(a) Isometric layout.



(b) Random layout.



(c) Circular layout.



(d) Force-directed layout.

Figure 2.7: Different drawings of an abstract graph.

- Each connection of the graph is drawn as a straight line.

Subfigure 2.7b shows an example of graph generated by random layout algorithm.

Force-directed layout algorithm: This algorithm simulates a physical system in order to visualize a network. The algorithm is commonly used for graph plotting which can be found in literature such as [26] [27] [28] [29] [30]. The initial step of force-directed method is to lay all the vertices randomly on the display. After that, two kinds of forces are defined and used: the repulsive force and the attractive force. The repulsive force, analogous to electrical force used to push vertices away from each other, existed between each pair of vertices. In a graph $G(V, E)$, The repulsive force is define as:

$$f_r(u, v) = -CK^2/||x_u - x_v||, \text{where } u \neq v, u, v \in V \tag{2.6}$$

where K is a parameter as the optimal distance, C is the strength relationship between repulsive and attractive force and x_i is the current location of vertex i on the display. The attractive force, analogous to mechanical spring force used to pull vertices towards, exists between a vertex and its connected neighbors. The attractive force is defined as:

$$f_a(u, v) = ||x_u - x_v||^2/K, u \leftrightarrow v, u, v \in V \tag{2.7}$$

As a result, the force of vertex u can be represented as:

$$f(u) = \sum f_r(u, v)|_{u \neq v} + \sum f_a(u, v)|_{u \leftrightarrow v}, u, v \in V \tag{2.8}$$

These forces work together and create movements that make system converge to a balanced state. This final configuration is used to layout the graph data. Subfigure 2.7d shows an example of graph generated by force-directed layout algorithm.

In this dissertation, we will use the force-directed layout algorithm to exhibit the structure of lattice graph in TEA. Figure 2.8 shows an example of comparison between the physical map locations of a real power grids and their graph layouts generated by force-directed method. In this figure, the left sub graph is the layout of the Italy power grid, each vertex represents a bus in the power grid while each connection represents a transmission line connecting two buses in the grid. Meanwhile, right sub graph is the map of Italy power system. The critical shape defining vertices in the graph and the corresponding physical locations in the map are circled and marked in the figure. The generated graph is consistent with the map. For example, the positions of eight substations in the map are consistent with those in the generated graph. This example shows the force-directed drawing can reflect the topology feature of the graph.

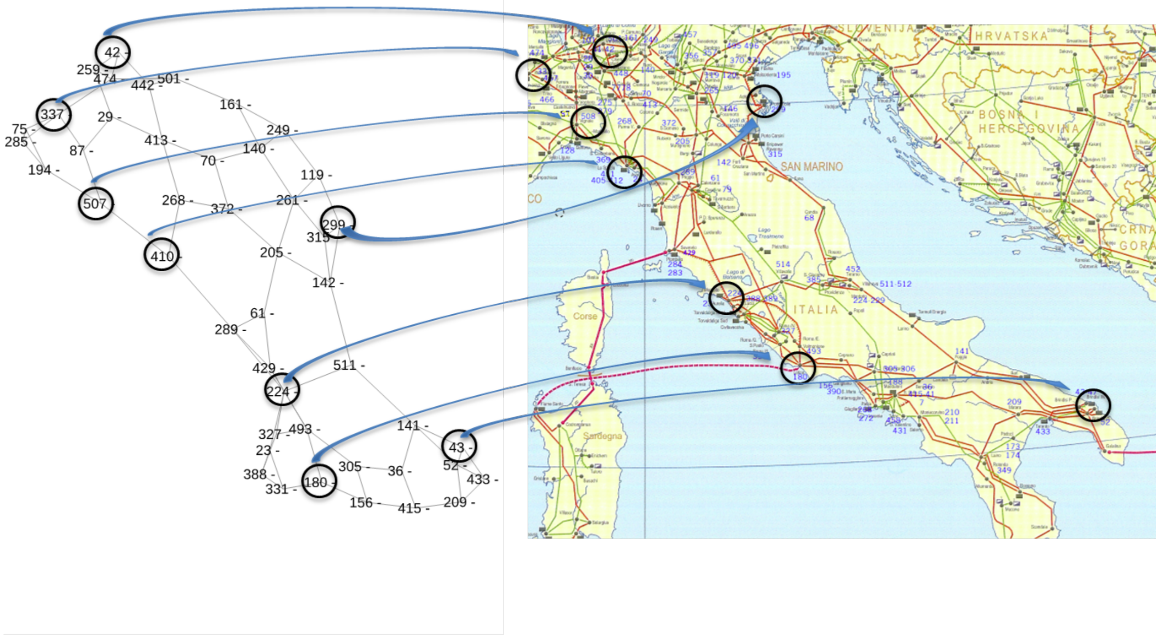


Figure 2.8: Comparison of force-directed layout method generated graph with the map of Italy power grid.

Chapter 3

Inspiration from Examples of Physics

In the previous chapter, we present the related concepts and methods of this dissertation. As mentioned, we want to design an automaton for analyzing the energy dynamics of a graph. For this, we imagine energy flow between two vertices to be analogous to the physical movement of one particle between two locations. So before introducing the mechanism of TEA, it is worthwhile to introduce the inspiration of the research. In this chapter, we will use particle movements as the example to show the idea behind this research.

This section consists of five parts: in section 3.1, Brownian motion will be introduced as an example to show the Langevin equation can be used to explain the stochastic motion of particles; from section 3.2 to section 3.4, the Lorenz system will be introduced as an example to show chaotic movements of particles can be synchronized by network. Section 3.2 exhibits the different chaotic motions of unsynchronized particles; section 3.3 introduces the chaotic motions of particles can be synchronized through connections; moreover, section 3.4 shows particles in the network can also be synchronized; finally in the section 3.5, we will summarize the whole chapter.

3.1 Implication of Langevin Equation on Free Single Particle

Brownian motion describes the stochastic movement of particle due to collisions with other small molecules[31]. Figure 3.1 [32] shows an example of Brownian motion. In this movement, the size of particle is considered vastly larger than the size of molecules. Thus, the motion of Brownian particle is vastly slower than the motion of molecules in the fluid. There are several different approaches for explaining Brownian motion, Langevin equation is one of them. In the following part of this section, we will use one-dimension Brownian motion as an example to show the idea of Langevin equation.

Figure 3.1: Stochastic motion of a Brownian particle.

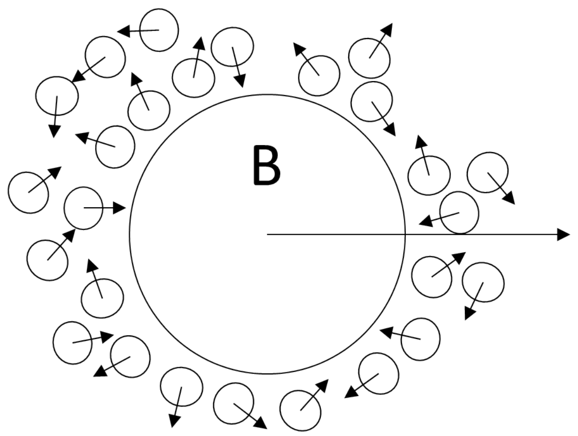


Figure 3.2: A large particle B immersed in a fluid of small molecules.

Langevin equation is a stochastic differential equation that explains the evolution of particle stochastic movements. Typically this equation divided the stochastic movements into macroscopic (slow) movements and microscopic (fast) movements, where the macroscopic movements change slowly compared to the microscopic movements. Thus, Langevin equation considers instantaneous force on a particle at time t consists of two types of forces, where one is in macroscopic scale while the other is in microscopic scale as Figure 3.2 shows. Moreover, Langevin equation in stochastic particle movement can be written as follows:

$$\begin{aligned} \frac{d(x(t))}{dt} &= v(t) \\ \frac{d(v(t))}{dt} &\propto \mathcal{F}(t) + F(t) \end{aligned} \tag{3.1}$$

In this equation:

- $\mathcal{F}(t)$ stands for the sum of the force drive slow action. In Brownian motion example, $\mathcal{F}(t)$ is the friction force of Brownian particle.
- $F(t)$ stands for the sum of the force drive fast action. In Brownian motion example, $F(t)$ is the stochastic force due to the collisions of Brownian particle with the surrounding small molecules. This force is assumed to be a Gaussian

process with

$$\langle F(t)\rangle = 0, \langle F(t_1)F(t_2)\rangle = g\delta(t_1 - t_2) \tag{3.2}$$

where $\langle ... \rangle$ implies the sample average, both of t_1 and t_2 are times, g is a measure of the strength of the fluctuating force and delta function indicates there is no correlation between two stochastic force at different time t_1 and t_2.

From this function, we can conclude that stochastic behavior of particle movement can be explained as the combination of the forces driven by slow action and fast action.

3.2 Chaotic Movements of Unsynchronized Particles

In the rest of this chapter, we will use Lorenz equation to simulate the trajectories of chaotic particle movements. The Lorenz equation is a group of nonlinear differential equations [33] [34] [35] [36] [37], which can be represented by spatial variables x y and z as follows:

$$\begin{aligned} \frac{dx}{dt} &= \sigma(y - x) \\ \frac{dy}{dt} &= x(\rho - z) - y \\ \frac{dz}{dt} &= xy - \beta z \end{aligned} \tag{3.3}$$

where t is the time, σ ρ β are the three input parameters. Figure 3.3 shows a sample solution of Lorenz equation.

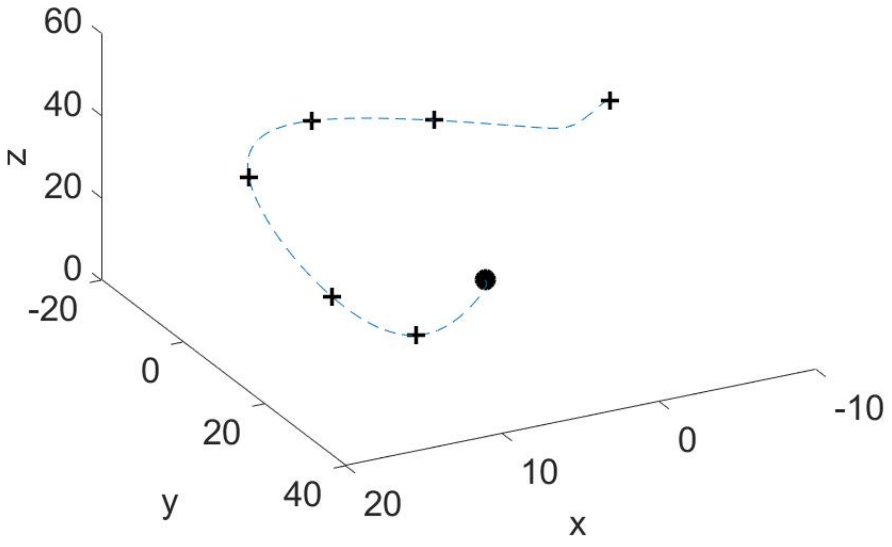


Figure 3.3: a sample solution of Lorenz equation.

Lorenz equation is notable for several natures, in this section we mainly discuss two of them:

- Lorenz equation has chaotic solution of certain parameters. Such as: for the input parameters $\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$, Figure 3.4 shows the chaotic solution of Lorenz equation with initial condition $(x_1(0) = 1, y_1(0) = 1, z_1(0) = 1)$ from time 0 to time 100.

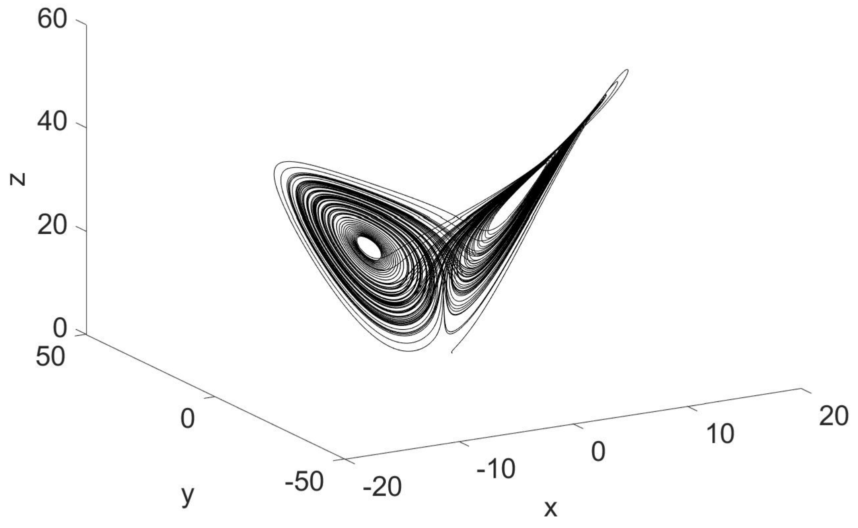


Figure 3.4: Solution of Lorenz equation with parameters $\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$ and initial condition $(x(0) = 1, y(0) = 1, z(0) = 1)$ for $0 \leq t \leq 100$.

- Lorenz equation is highly sensitive to initial conditions. Figure 3.5 shows an example: the x-coordinate trajectory of the solution modeled by identical Lorenz equations with initial conditions $(x(0) = 1, y(0) = 1, z(0) = 1)$ is different from the x-coordinate trajectory of another solution modeled by the same Lorenz equations with different initial conditions $(x(0) = 1.001, y(0) = 1, z(0) = 1)$. The two trajectories remain close from time 0 to time 20, then start to exhibit different chaotic behaviors. In this section, we only discuss trajectory of the x-coordinate for simplicity since the trajectory of the y- and z-coordinate behaves in the same manner as that of the x-coordinate.

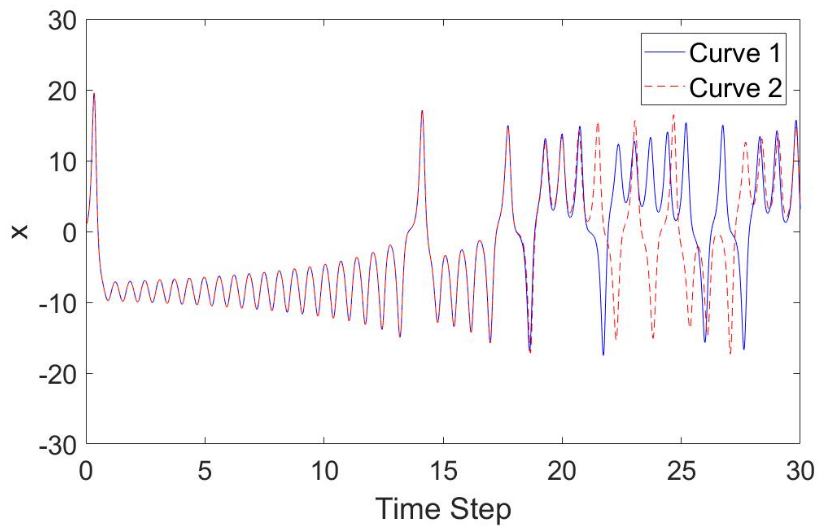


Figure 3.5: Sensitivity of particles to the initial condition. Curve 1 is the trajectory of solution with initial condition $(x(0) = 1, y(0) = 1, z(0) = 1)$ for $0 \leq t \leq 30$; curve 2 is the trajectory of solution with initial condition $(x(0) = 1.001, y(0) = 1, z(0) = 1)$ for $0 \leq t \leq 30$.

These two natures show that given two separate particles with the same Lorenz equation but slightly different initial conditions, they will exhibit different chaotic trajectories without synchronization.

3.3 Chaotic Movements of Synchronized Particles

In this section we will illustrate that the chaotic motions of different particles can be synchronized through connections among them.

3.3.1 Synchronization of Particles Movements through One Connection

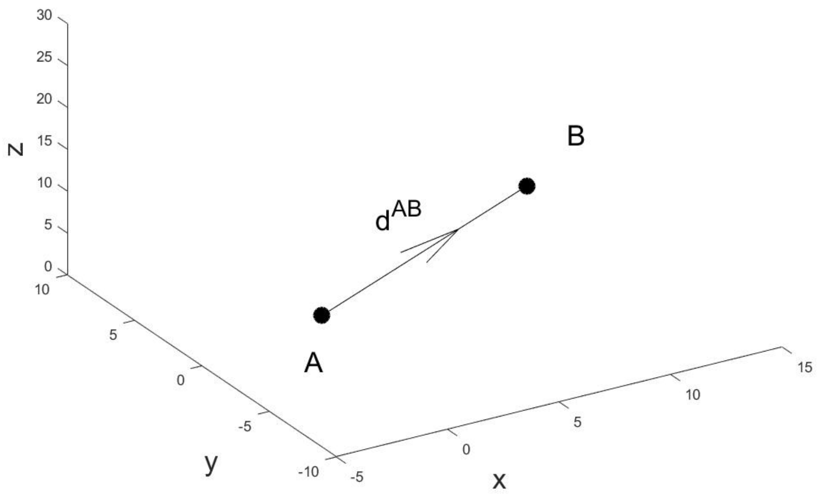


Figure 3.6: Two particles coupled by a connection.

Figure 3.6 shows an example of two particles coupled through a connection. Particle A and particle B are modeled by Lorenz equation with same parameters[38]; the two particles start at two different locations, $(1, 1, 1)$ and $(12, 4, 6)$; the two particles connected through a connection which leads to the movement of particle A affects the movement of particle B; we suppose two particles are linearly coupled. Thus, we represent these two particle movements by the following Lorenz equations:

$$\begin{aligned}\frac{dx_A}{dt} &= \sigma(y_A - x_A)\\ \frac{dy_A}{dt} &= x_A(\rho - z_A) - y_A\\ \frac{dz_A}{dt} &= x_A y_A - \beta z_A\end{aligned} \tag{3.4}$$

for the movement of particle A and

$$\begin{aligned}
\frac{dx_B}{dt} &= \sigma(y_B - x_B) + d_x^{AB} * (x_A - x_B) \\
\frac{dy_B}{dt} &= x_B(\rho - z_B) - y_2 + d_y^{AB} * (y_A - y_B) \\
\frac{dz_B}{dt} &= x_B y_B - \beta z_2 + d_z^{AB} * (z_A - z_B)
\end{aligned} \tag{3.5}$$

for the movement of particle B. In this example, the coupling of particle A and particle B is represented by d_x^{AB} d_y^{AB} d_z^{AB}, which are called as coupling coefficients. Suppose the system parameters for particle A and particle B are $\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$ and coupling coefficients $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1.5$, the simulation results can be seen as Figure 3.7 shows: the trajectories of particle A and particle B exhibit different chaotic behavior from time 0 to time around 3, and later become closely to each other. We can conclude that trajectory of particle B is synchronized with the trajectory of particle A.

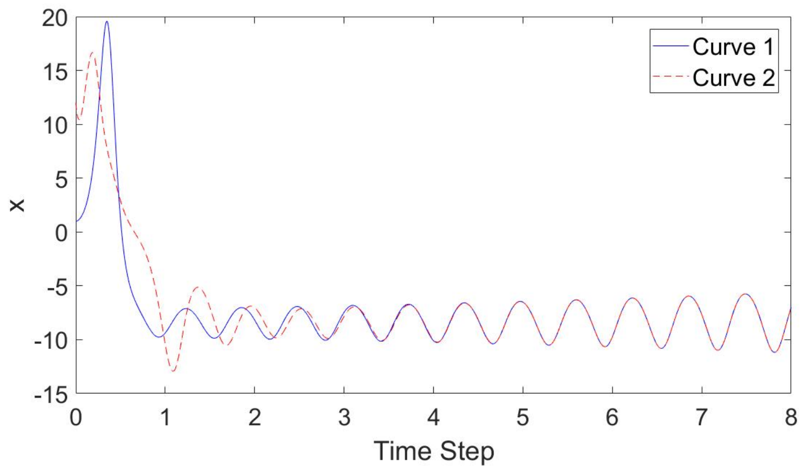


Figure 3.7: Synchronization of particle movements in two-particles one-connection network with coupling coefficients $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1.5$ in $0 \leq t \leq 8$. Curve 1 is the movement trajectory of particle A; Curve 2 is the movement trajectory of particle B.

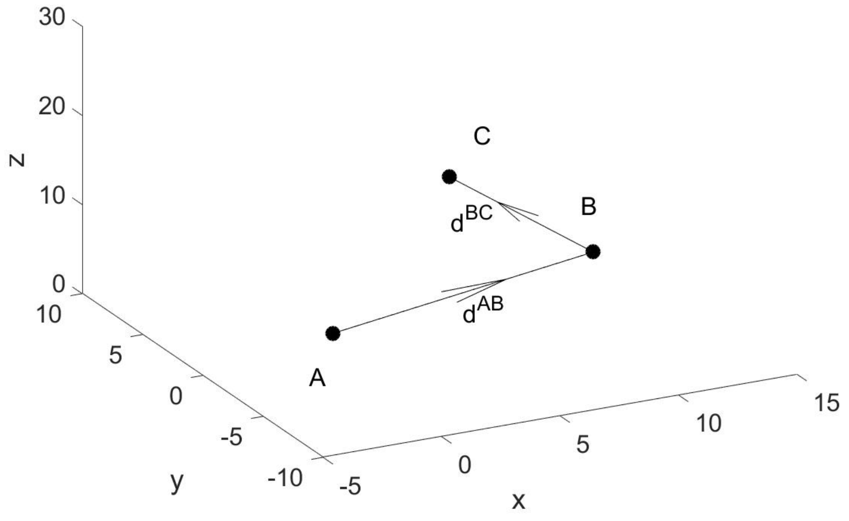


Figure 3.8: An example of three-particle and two-connection radial connected system.

3.3.2 Synchronization of Particles Movements through Radial Connections

Figure 3.8 shows an example of radial connected system: this example consists of three particles particle A, particle B and particle C; each of those particles is modeled by Lorenz equations with same parameters; the three particles starts at three different locations, $(1, 1, 1)$ $(12, 1, 5)$ and $(10, 9, 7)$; the two connections in the network lead to the movement of particle A affects that of particle B and the movement of particle B affects that of particle C; the same as previous example, we suppose those particles are linearly coupled. Thus, we represent these three particle movements by the following Lorenz equations:

$$\begin{aligned} \frac{dx_A}{dt} &= \sigma(y_A - x_A) \\ \frac{dy_A}{dt} &= x_A(\rho - z_1) - y_A \\ \frac{dz_A}{dt} &= x_A y_A - \beta z_A \end{aligned} \tag{3.6}$$

for the movement of particle A,

$$\begin{aligned}\frac{dx_B}{dt} &= \sigma(y_B - x_B) + d_x^{AB} * (x_A - x_B)\\ \frac{dy_B}{dt} &= x_B(\rho - z_B) - y_B + d_y^{AB} * (y_A - y_B)\\ \frac{dz_B}{dt} &= x_B y_B - \beta z_B + d_z^{AB} * (z_A - z_B)\end{aligned} \tag{3.7}$$

for the movement of particle B, and

$$\begin{aligned}\frac{dx_C}{dt} &= \sigma(y_C - x_C) + d_x^{BC}(x_B - x_C)\\ \frac{dy_C}{dt} &= x_C(\rho - z_C) - y_C + d_y^{BC}(y_B - y_C)\\ \frac{dz_C}{dt} &= x_C y_C - \beta z_C + d_z^{BC}(z_B - z_C)\end{aligned} \tag{3.8}$$

for the movement of particle C. The effect of particle A on particle B is given by coupling coefficients, d_x^{AB}, d_y^{AB}, d_z^{AB} while the effect of particle B on particle C is given by coupling coefficients, d_x^{BC}, d_y^{BC}, d_z^{BC}. Suppose parameters are $\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$ and the coupling coefficients $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1.5$, $d_x^{BC} = d_y^{BC} = d_z^{BC} = 1.5$, the simulation result at x-coordinate from time=0 to time=8 of this radial connected system can be seen in Figure 3.9: the movement trajectories of particle B and particle C exhibit different chaotic behaviors at the beginning, and later become synchronized to the movement trajectory of particle A. The time lag for movement of particle B synchronized with the movement of particle A is shorter than that between particle C and particle A. This example shows that the connections are capable of synchronizing the particle movements in the radial connected system.

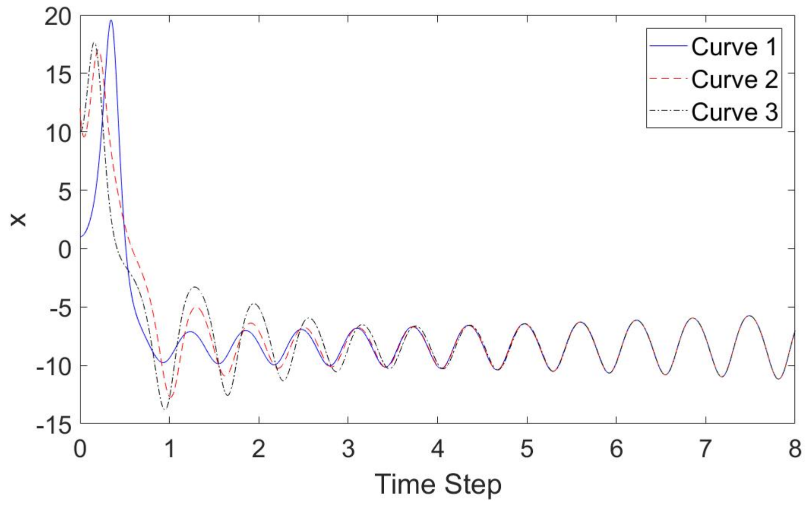


Figure 3.9: Synchronization of particle movements in three-particle two-connection network with coupling coefficients $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1.5$, $d_x^{BC} = d_y^{BC} = d_z^{BC} = 1.5$ for $0 \leq t \leq 8$. Curve 1 is the movement trajectory of particle A; curve 2 is the movement trajectory of particle B; curve 3 is the movement trajectory of particle C.

3.4 Impacts of Coupling Structure on Synchronization of Particles Movements

In this section we will illustrate how the synchronized motions of particles in the network can be influenced by coupling structure.

3.4.1 Synchronization of Particles Movements by Radial Coupling

We start the discussion by considering a simple networking in Figure 3.10. Suppose two particle A and B represent by 3.4 and 3.5 with parameter $\sigma = 10$ $\rho = 28$ $\beta = \frac{8}{3}$, and they start at two different locations $(1, 1, 1)$ and $(12, 4, 6)$. By testing three different sets of coupling coefficients d_x^{AB}=d_y^{AB}=d_z^{AB} $= 0.75$, d_x^{AB}=d_y^{AB}=d_z^{AB} $= 1.5$ and d_x^{AB}=d_y^{AB}=d_z^{AB} $= 5$, we can conclude the movement trajectory of particle B under three different coupling coefficients vary with each other. As Figure 3.11 shows, when increasing the coupling coefficients d_x^{AB} d_y^{AB} d_z^{AB}, the time lag for movement of

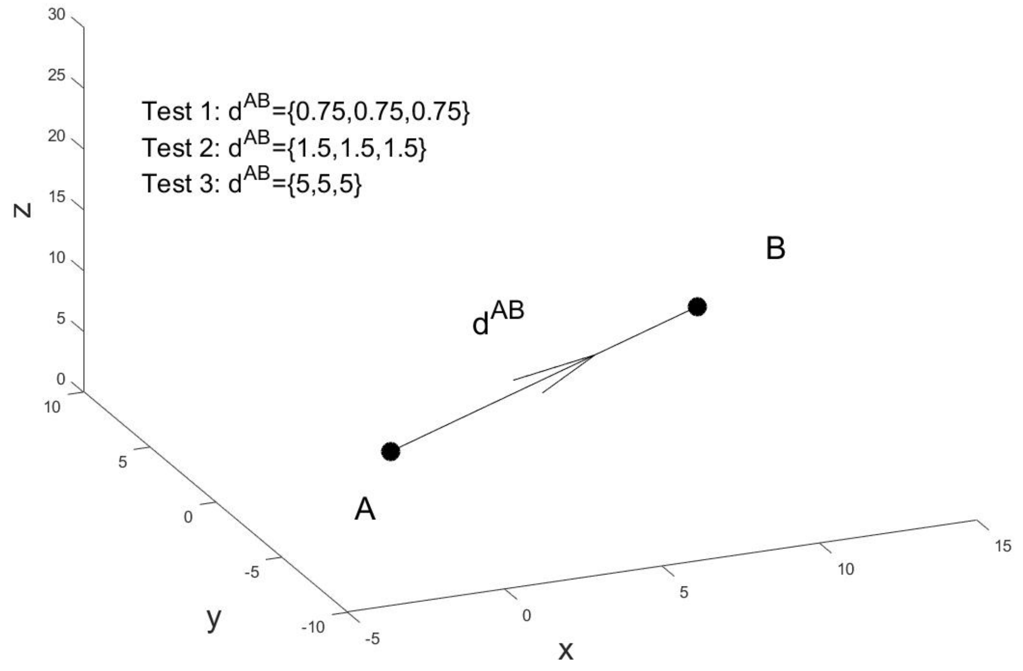


Figure 3.10: Two particles coupled by a radial connection with three different coupling coefficients.

particle B synchronized with movement of particle A decreased.

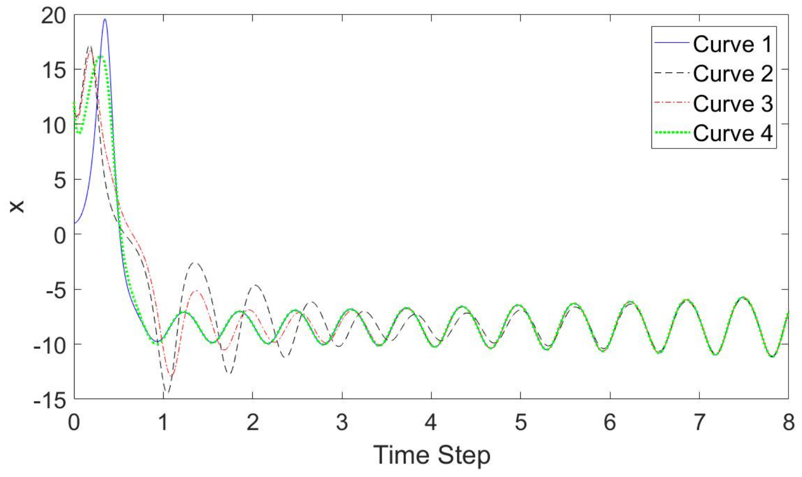


Figure 3.11: Behaviors of movement of particle B synchronized with movement of particle A upon different coupling coefficients for $0 \leq t \leq 8$. Curve 1 is the movement trajectory of particle A; Curve 2 is the movement trajectory of particle B coupled by $d_x^{AB} = d_y^{AB} = d_z^{AB} = 0.75$; Curve 3 is the movement trajectory of particle B coupled by $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1.5$; Curve 4 is the movement trajectory of particle B coupled by $d_x^{AB} = d_y^{AB} = d_z^{AB} = 5$.

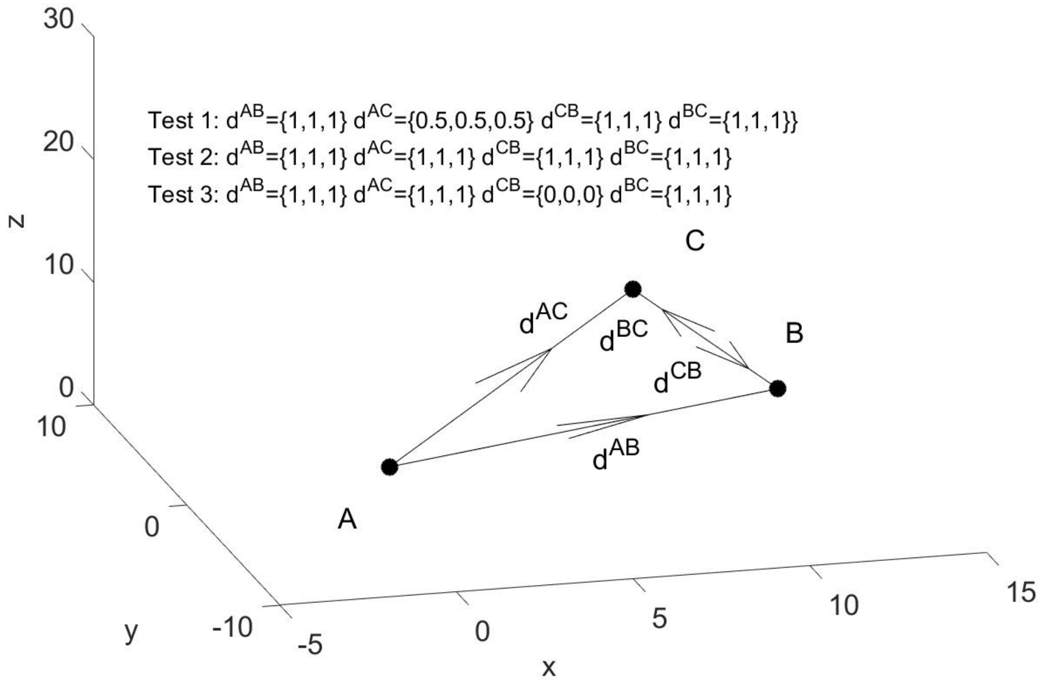

Test 1: d^{AB}={1,1,1} d^{AC}={0.5,0.5,0.5} d^{CB}={1,1,1} d^{BC}={1,1,1}}
Test 2: d^{AB}={1,1,1} d^{AC}={1,1,1} d^{CB}={1,1,1} d^{BC}={1,1,1}
Test 3: d^{AB}={1,1,1} d^{AC}={1,1,1} d^{CB}={0,0,0} d^{BC}={1,1,1}
A
B
C
d^{AB}
d^{AC}
d^{BC}
d^{CB}
x
y
z

Figure 3.12: A looped network with three different sets of coupling coefficients.

3.4.2 Synchronization of Particles Movements by Looped Coupling

Furthermore, we also consider the synchronization of particle movements in a three-particle and four-connection looped network on different coupling coefficients. Based on previous three-particle and two-connection radial connected system in Figure 3.8, we add a connection from particle A to particle C and a connection from particle C to particle B as Figure 3.12 shows. Those connections lead to the movement of particle A affects that of particle B, and the movements of particle C and the particle B affect with each other. We use the following Lorenz equations to represent these three particle movements:

$$\begin{aligned}
\frac{dx_A}{dt} &= \sigma(y_A - x_A) \\
\frac{dy_A}{dt} &= x_A(\rho - z_A) - y_A \\
\frac{dz_A}{dt} &= x_A y_A - \beta z_A
\end{aligned} \tag{3.9}$$

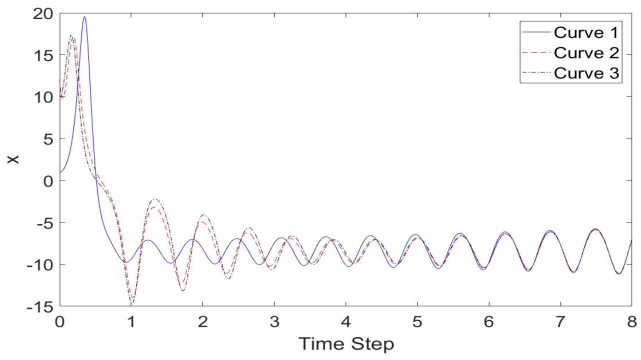
for the movement of particle A,

$$\begin{aligned}
\frac{dx_B}{dt} &= \sigma(y_B - x_B) + d_x^{AB} * (x_A - x_B) + d_x^{CB}(x_C - x_B) \\
\frac{dy_B}{dt} &= x_B(\rho - z_B) - y_B + d_y^{AB} * (y_A - y_B) + d_y^{CB}(y_C - x_B) \\
\frac{dz_B}{dt} &= x_B y_B - \beta z_B + d_z^{AB} * (z_A - z_B) + d_z^{CB}(z_C - z_B)
\end{aligned} \tag{3.10}$$

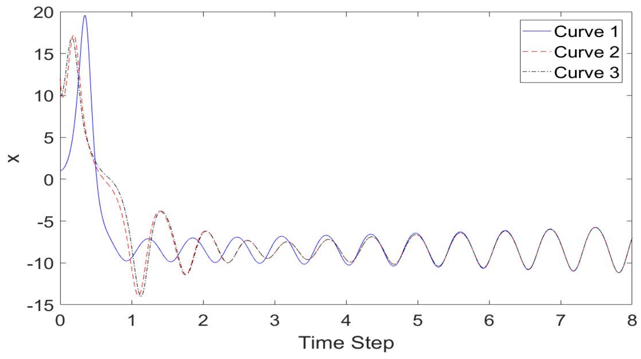
for the movement of particle B, and

$$\begin{aligned}
\frac{dx_C}{dt} &= \sigma(y_C - x_C) + d_x^{AC}(x_A - x_C) + d_x^{BC}(x_B - x_C) \\
\frac{dy_C}{dt} &= x_C(\rho - z_C) - y_C + d_y^{AC}(y_A - y_C) + d_y^{BC}(x_B - x_C) \\
\frac{dz_C}{dt} &= x_C y_C - \beta z_C + d_z^{AC}(z_A - z_C) + d_z^{BC}(x_B - x_C)
\end{aligned} \tag{3.11}$$

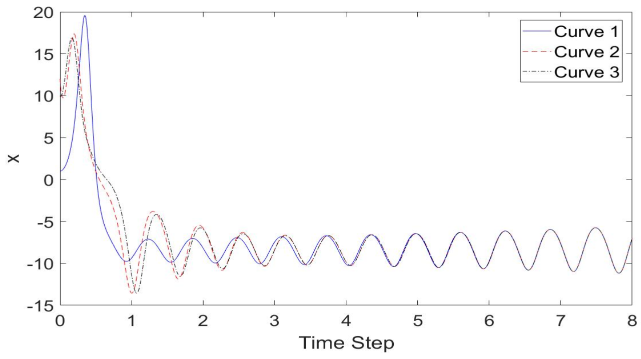
for the movement of particle C, where the two sets of parameters, $d_x^{AC}{=}d_y^{AC}{=}d_z^{AC}$ and $d_x^{BC}{=}d_y^{BC}{=}d_z^{BC}$, stand for coupling coefficients of movement of particle A affects on that of particle C and movement of particle B affects on that of particle C. In Figure 3.13, solid curves are the movement trajectory of particle A, the dashed curves are the movement trajectory of particle B and dash-dot curve are the movement trajectory of particle C; both the movements of particle B and particle C will become synchronized with movement of particle A. Depending on different coupling coefficients, the movements of particle B and particle C show different features; in (a), all of the three movements become synchronized at about $t = 6$; in (b) and (c), all of the three movements become synchronized at about $t = 4$; the movements of particles B and C show different trajectories from $t = 1$ to $t = 4$. This example shows that by different coupling rules, a simple particles connection is capable of producing a great variety of unexpected synchronization behaviors.



(a) Trajectory of movements of particles by coupling coefficients: $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1, d_x^{AC} = d_y^{AC} = d_z^{AC} = 0.5, d_x^{CB} = d_y^{CB} = d_z^{CB} = 1, d_x^{BC} = d_y^{BC} = d_z^{BC} = 1$



(b) Trajectory of movements of particles by coupling coefficients: $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1, d_x^{AC} = d_y^{AC} = d_z^{AC} = 1, d_x^{CB} = d_y^{CB} = d_z^{CB} = 1, d_x^{BC} = d_y^{BC} = d_z^{BC} = 1$



(c) Trajectory of movements of particles by coupling coefficients: $d_x^{AB} = d_y^{AB} = d_z^{AB} = 1, d_x^{AC} = d_y^{AC} = d_z^{AC} = 1, d_x^{CB} = d_y^{CB} = d_z^{CB} = 0, d_x^{BC} = d_y^{BC} = d_z^{BC} = 1$

Figure 3.13: Synchronization of particle movements in three-particle four-connection network by different coupling coefficients for $0 \leq t \leq 8$. In subfigure (a)-(c), curve 1 is the movement trajectory of particle A; curve 2 is the movement trajectory of particle B; curve 3 is the movement trajectory of particle C.

3.5 Summary

In this chapter, we introduce the inspiration of this research. First we explained the stochastic motion of particle can be represented as a way of Langevin equation, which is a combination of macroscopic (slow) movement and microscopic (fast) movement.

Next, we use Lorenz system as an example to exhibit the chaotic motions of particles can be synchronized by connections, and moreover by different coupling structures. By proposing particular cases of networks, we find: the movements of particles can become synchronized by links or even by a connected network; by changing simple coupling rules, the particles in the connected network are capable of producing a great variety of unexpected synchronization behaviors.

In the next chapter, we will the hints from this chapter to form the Transient Energy Automaton model.

Chapter 4

Proposition of Transient Energy Automaton

Transient Energy Automaton(TEA) is a mathematical model to analyze the energy flow on the network once triggered by disturbances. The graph of TEA consists of a collection of vertices. Each vertex has a value at a time, which stands for the energy distributed on the vertex. The configuration of TEA is represented by the values of all vertices on the graph and which can be seen as the energy distribution on the graph. The vertices update their values synchronously on discrete time steps according to the rules. The new value of each vertex depends on the previous configuration of TEA. The TEA evolution is composed of configuration changing over time.

The differences between TEA and CAs are: 1) TEA considers all vertices in the graph do the same action, which is in either fast action or slow action, at a time; 2) TEA assumes each vertex interacts globally with all of the other vertices on the graph; 3) TEA considers the energy distribution at next time step depends on that of current time; 4) The configuration of TEA is the energy distribution on the graph at a time. In this section, we will introduce the general description of TEA. For simplify, we discuss the TEA with two-dimensional graph, particularly lattice graphs, in this dissertation.

4.1 Automation Mechanism

We start this chapter by introducing the mechanism of TEA. We assume each vertex in the TEA is capable of processing energy by either reflecting or absorbing. When a vertex receives energy from the others, this vertex has a tendency to absorb and reflect the received energy. To simplify, we suppose that one vertex will absorb part of the received energy while reflecting the rest part of the received energy to the other vertices. Therefore, we define two actions, *fast action* and *slow action*, to present the process of energy reflection and energy absorption.

Firstly, we define the fast and slow actions in the TEA as follows:

Definition 1. Fast Action: A process for the vertex to reflect energy. We suppose after the vertex receives energy, part of the received energy will be reflected from the vertex and received by other vertices on the graph. This process takes a relatively short time when compared with slow action.

Definition 2. Slow Action: A process for the vertex to absorb energy. We suppose after the vertex receives energy, part of the received energy will be absorbed by this vertex and the rest of them will be processed by fast action. This process takes a relatively long time when compared with fast action.

Secondly, we define the governing rules for TEA. Since the fast and slow actions of each vertex are coupled with each other, the two actions for each vertex requires complex rules to govern them. In order to simplify and implement those actions into TEA model, we simplify these rules as follows:

- The fraction of energy that can be absorbed by each vertex in the slow action depends on the capability of the vertex. We define this fraction is a constant, which we call as absorption ratio;

- For each vertex, the amount of energy received at a local site (from the disturbance vertex or the reflected energy from neighboring vertices) in the fast action is a function of the graph geometry including the structure of lattice and value configuration of edges. We define this function is determined by a constant, which we call as dispersion ratio;

- We suppose for all of the vertices in TEA, the fast action and slow action are synchronized. For each discrete time step, there contains one fast action and one slow action;

- Such rules are functions of the geometry and value configuration of graph and the property of vertices, and ultimately enable the TEA to create time evolution of energy distributions that may affect the dynamics of the system.

In other words, the first rule shows the absorption ratio is given by the property of that vertex, which is assumed to be known; the second rule shows dispersion ratio is related to graph geometry; the third rule shows all the vertices synchronize their actions; the fourth rule summarizes how to create the evolution of TEA.

In the rest of this section, we will introduce the approach of calculating dispersion ratio. First, we consider the graph of TEA defined by an undirected, weighted and connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N vertices. The set of vertices is given by $\mathcal{V} = \{i : i = 0, ..., N-1\}$, and the set of connections is given by $\mathcal{E} = \{(i, j) : i, j \in \mathcal{V}, i \neq j\}$. Furthermore, we define the shortest distance between each pair of vertices is given by the mapping function 4.1:

$$B : \mathcal{E} \to \mathbb{R} \tag{4.1}$$

The graph can then be described by $\mathcal{N} = (\mathcal{V}, \mathcal{E}, B)$. We have the shortest distance matrix $A \in \mathbb{R}^{N\times N}$ as follows:

$$A_{ij} = \begin{cases} B_{ij} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

where B_{ij} can be any kind of shortest distance between i and j.

Next, we introduce the connection strength matrix of the graph $\mathcal{N}$, which is labeled as T. In this part, the term *connection strength* represents the ability of energy exchange between a pair of vertices: the stronger the strength between two vertices, the higher the intensity of energy exchange between them. Mathematically, for the vertex i and vertex j with connection strength $T_{ij} \in T$, a greater distance between them would represent weaker connection strength and hence a smaller value of T_{ij}. Particularly, we define the connection strength matrix $T \in \mathbb{R}^{N\times N}$ as follows:

$$T_{ij} = \begin{cases} (\frac{1}{2})^{A_{ij}} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

Remark 1. *We suppose in fast action, there is no energy dispersion or consumption during the procedure of energy exchange.*

Remark 2. *We suppose a vertex cannot receive the energy reflected from itself.*

Connection strength matrix T helps us in calculating energy distribution by fast action which will explain by using the following example. Assume N-vertex graph $\mathcal{N} = \{\mathcal{V}, \mathcal{E}, B\}$ with connection strength matrix $T \in \mathbb{R}^{N\times N}$, the vertex i will reflect and distribute the entire energy p^i by fast action to the graph. Using T, we can calculate p^i_j ($j \in \mathcal{V}$ and $j \neq i$), which means the energy reflected from vertex i and received by vertex j, is

$$p^i_j = p^i \times \frac{T_{ij}}{\sum_{k=0,k\neq i}^{N-1} T_{ik}} \tag{4.4}$$

Assuming $P^i \in \mathbb{R}^{N \times 1}$ to describe the energy reflected from vertex i to all vertices, we have:

$$P^i = \begin{bmatrix} p_0^i \\ p_1^i \\ \vdots \\ p_{N-1}^i \end{bmatrix} \tag{4.5}$$

where p_i^i is 0 (by remark 2). We denote the following function to describe the mapping $p^i \times T \rightarrow P^i$

$$D : InjectE \times CSMatrix \rightarrow Dist(InjectE) \tag{4.6}$$

where $InjectE$ stands for injected energy, $CSMatrix$ stands for the connection strength matrix and $Dist(InjectE)$ stands for the distribution of injected energy. We denote this function as *Energy Distribution Function* (EDF). Energy distribution function D is important in TEA and we will use this function in the following discussion.

Before we continue, let me introduce a notation that will be helpful in following discussion. We label a matrix M whose elements of ith row and ith column are all 0 as $M_{-i,-i}$, i.e. the matrix M is

$$M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \tag{4.7}$$

and the corresponding matrix $M_{-3,-3}$ is

$$M_{-3,-3} = \begin{bmatrix} 1 & 2 & 0 & 4 \\ 5 & 6 & 0 & 8 \\ 0 & 0 & 0 & 0 \\ 13 & 14 & 0 & 16 \end{bmatrix} \tag{4.8}$$

4.2 The Model

In the previous section, we assume all the actions of vertices in TEA are synchronized with a global clock and executed at the same time. Moreover, we start by using two states to describe the two actions. They are:

- -1: The vertex is in fast action.
- 1: The vertex is in slow action.

Secondly, we introduce the initial condition of TEA. We use the term *Start vertex* to indicate a vertex where energy is injected into the graph at the very first time. We assume that after initial step, the start vertex cannot receive or keep any energy but it still keep on connected in the network (or more precisely, the network connection structure is not changed). Thus, we can define start vertex as:

Definition 3. Start vertex: Start vertex is the vertex that receives the injected energy and reflects all of its received energy to the other vertices. This is defined as the initial step. After this step, start vertex will remain connected to the graph without absorbing or receiving energy.

After the energy is injected into start vertex, the energy evolution in TEA begins.

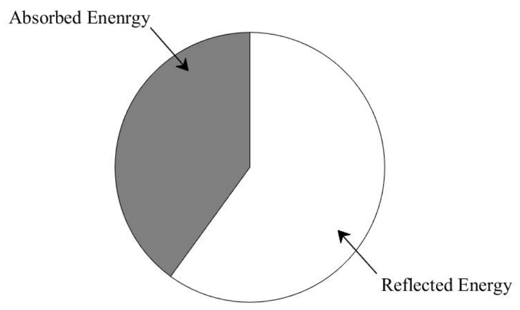


Figure 4.1: The received energy can be divided into two parts, which are absorbed energy and reflected energy.

Thirdly, we will focus on iterations of TEA. As we assume in the previous section, there exists a fast action and a slow action for each time step from time t_1 to time t_1+1 where $t_1 = 0, 1, ...$ The fast action takes place from time t_1 to time $t_1+\epsilon_1$ while the slow action takes place from time $t_1+\epsilon_1$ to time t_1+1 where $\epsilon_1 \ll 1$. In order to help us illustrate energy flow caused by slow action and fast action, we define the concepts of *absorbed energy* and *reflected energy* as Figure 4.1 shows. For the received energy of each vertex on graph, it can be divided into two parts, they are:

Definition 4. Absorbed energy: It is the absorbed parts of received energy. We denote it as $A^i(t_1)$ where $t_1 = 0, 1, ...$ is the time step and i is the vertex. This part of the energy will be absorbed by the slow action of the vertex.

Definition 5. Reflected energy: It is the reflected parts of received energy. We denote it as $R^i(t_1)$ where $t_1 = 0, 1, ...$ is time step and i is the vertex. This part of the energy will be reflected by the fast action of the vertex.

The ratio between absorbed energy and received energy is defined as the property of vertex. We use *reflection ratio* to represent the ratio between reflected energy and received energy.

Definition 6. Reflection ratio: It is a ratio between reflected energy and received energy of a vertex. In this dissertation, we use r_i to represent the reflection ratio of vertex i. Unlike absorption ratio, this ratio stands for the ability of the vertex to reflect energy.

Besides, we also define the concept of *Total contained energy* and *Entire absorbed energy*

Definition 7. Total contained energy: It is the entire energy contained in the vertex i at time $t_1 - 1$. We denote it as $E^i(t_1)$.

Definition 8. Entire absorbed energy: It is the entire energy absorbed by the vertex i till current time.

Formally, the TEA can be defined as a structure below:

Definition 9. A Transient Energy Automaton is a 8-tuples $\mathcal{A} = (\mathcal{N}, T, r, v_0, \pi, \mathcal{S}, \Sigma, \tau)$ where:

- $\mathcal{N} = \{\mathcal{V}, \mathcal{E}, B\}$ is an undirected graph of the $\mathcal{A}$. The vertices set $\mathcal{V}$ of this graph has N elements, B is the mapping function to obtain the shortest distance between each pair of vertices;
- $T \in \mathbb{R}^{N \times N}$ is the connection strength matrix of the graph;
- $r \in \mathbb{R}^{N \times 1}$ is the reflection ratio vector of the graph;
- $v_0 \in V(\mathcal{G})$ is the start vertex;
- $\pi \in \mathbb{R}$ is the entire energy injected into vertex v_0;
- $\mathcal{S}$ is the nonempty set of vertices states at a time. The state for vertex i at time t is

$$\mathcal{S}_i(t) = \begin{cases} -1 \quad when \quad t_1 \leq t \leq t_1 + \epsilon_1 \\ 1 \quad when \quad t_1 + \epsilon_1 \leq t \leq t_1 + 1 \end{cases} \tag{4.9}$$

where $\epsilon_1 \ll 1$, -1 stands for fast action and 1 stands for slow action.

- Σ is a triplet energy values set where $\Sigma(t_1) = (R(t_1),\ A(t_1),\ E(t_1))$:
  - $R \in \mathbb{R}^{N\times 1}$ stands for the reflected energy for each vertex at the time $t = t_1$, where:

$$R(t_1) = \begin{bmatrix} R^0(t_1) \\ R^1(t_1) \\ \vdots \\ R^{N-1}(t_1) \end{bmatrix} \tag{4.10}$$

  - $A \in \mathbb{R}^{N\times 1}$ stands for absorbed energy for each vertex at the time $t = t_1$, where:

$$A(t_1) = \begin{bmatrix} A^0(t_1) \\ A^1(t_1) \\ \vdots \\ A^{N-1}(t_1) \end{bmatrix} \tag{4.11}$$

  - $E \in \mathbb{R}^{N\times 1}$ stands for the total contained energy for each vertex at a time $t = t_1$, where:

$$E(t_1) = \begin{bmatrix} E^0(t_1) \\ E^1(t_1) \\ \vdots \\ E^{N-1}(t_1) \end{bmatrix} \tag{4.12}$$

- τ is a function that maps the current triplet energy values set into the next triplet energy values.

$$\Sigma(t_1 + 1) = \tau(\Sigma(t_1)) \tag{4.13}$$

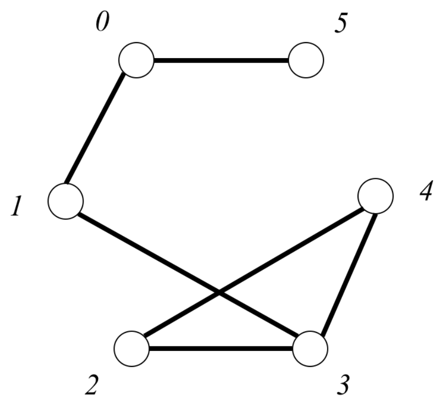


Figure 4.2: A six-vertex TEA graph

In the rest part of this section we will use an example of six-vertex connected graph in Figure 4.2 as to show the operations of TEA. In this example, the start vertex is vertex 0. When dynamics of energy flow triggered, the energy is reflected from start vertex to other vertices as Figure 4.3 shows. After initiation, the start vertex never keep any energy as Figure 4.4 shows. Figure 4.5 shows the energy flow of fast action on all of vertices except start vertex.

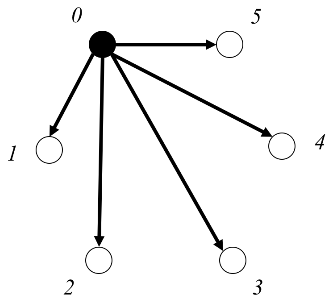


Figure 4.3: The energy flow of six-vertex graph at time $t = 0$: the energy disperse from start vertex 0 to other vertices by fast action.

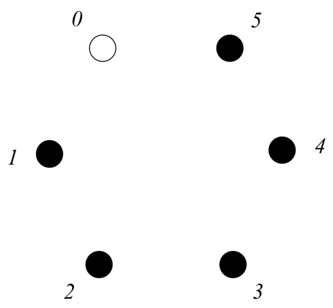
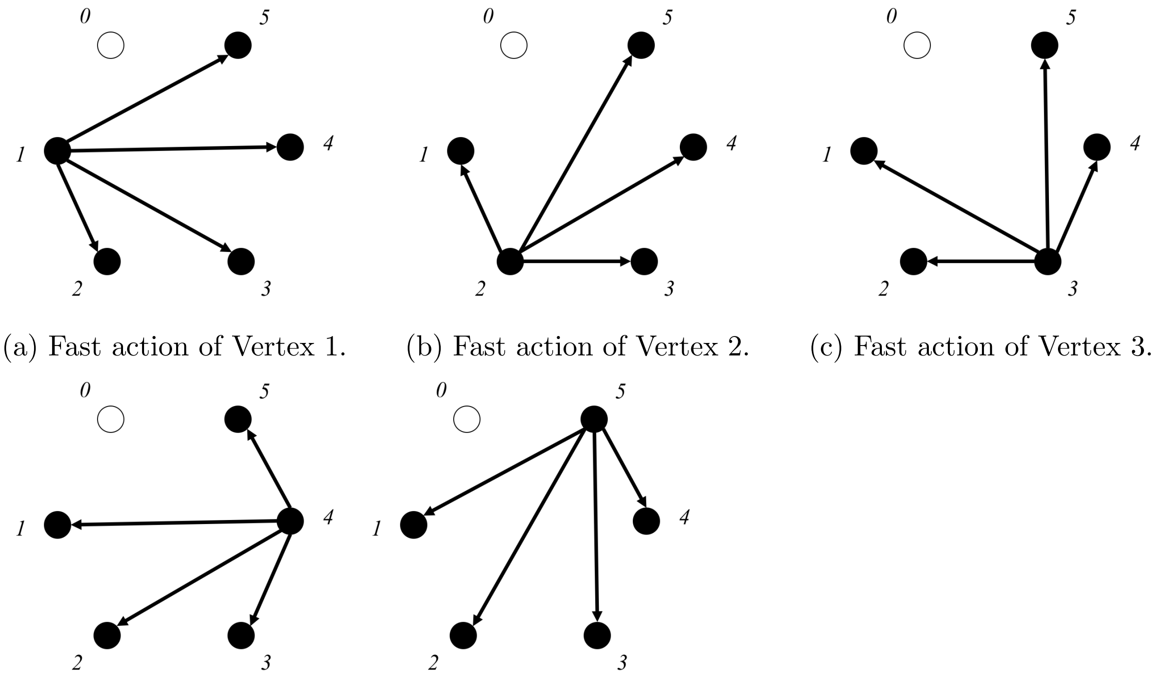


Figure 4.4: The energy distribution after initialization.



(a) Fast action of Vertex 1. (b) Fast action of Vertex 2. (c) Fast action of Vertex 3.

(d) Fast action of Vertex 4. (e) Fast action of Vertex 5.

Figure 4.5: The fast action of each vertex at time $t = t_1$.

4.3 Working Process

In the previous section, we introduced the model of TEA. The overall energy flow on the $G(V, E)$ is updated as Figure 4.6 shows: the rows of circular node stand for N vertices (from 0 to $N - 1$) while the columns stand for the discrete time (from 0 to $t_1 + 1$), i.e. the row v_0 and column 1 means the vertex v_0 at time 1; the arrows stand for the directions of energy flow, i.e. the arrow from row v_0 at column 0 to row 0 at column 1 means energy reflected from vertex v_0 at time 0 and received by vertex 0 at time 1. This graph consists of N-vertex and with connection strength matrix T. Initially, the energy flow from start vertex v_0 to other vertices except v_0. After initiation, for every vertex in the network except v_0, the energy will flow from this vertex to the other vertices except v_0.

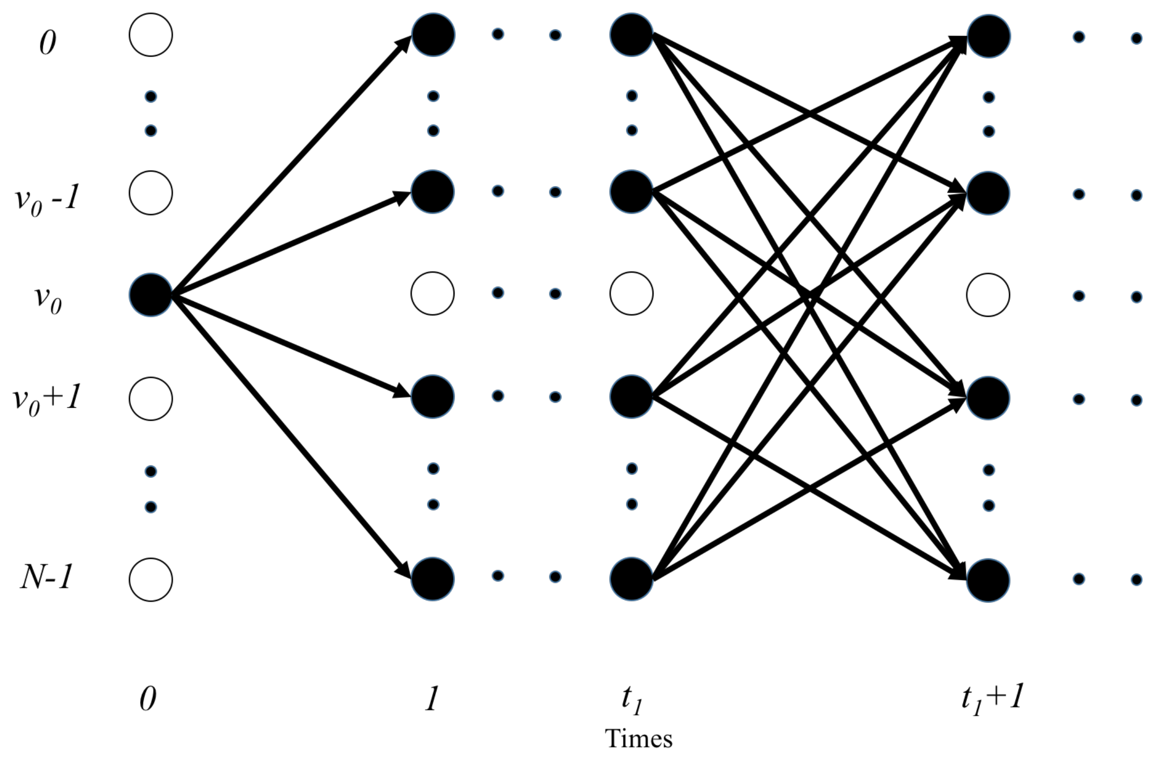


Figure 4.6: Energy flow in the graph.

Firstly, we discuss the energy flow at the initial step. Suppose the connection strength matrix is T, the energy flow from start vertex v_0 to other vertex i is determined by EDF(equation 4.6). We use $\alpha_i^{v_0}$ indicate $\frac{T_{v_0 i}}{\sum_{j=0,j\neq i}^{N-1} T'_{v_0 j}}$, then energy received by vertex i from start vertex is $E^{v_0} * \alpha_i^{v_0}$. The reflected energy for vertex i at $t = 0$ can be represent as:

$$R^i(0) = E^{v_0} * \frac{T_{v_0 i}}{\sum_{j=0,j\neq i}^{N-1} T_{v_0 j}} * \gamma^i = E^{v_0} * \alpha_i^{v_0} * \gamma^i \tag{4.14}$$

where E_{v_0} stands for the energy injected to the start vertex. The absorbed and total contained energy for vertex i, where $i \neq v_0$, at time $t = 0$ are

$$A^i(0) = E^{v_0} * \frac{T_{v_0 i}}{\sum_{j=0, j\neq i}^{N-1} T_{v_0 j}} * (1 - \gamma^i) = E^{v_0} * \alpha_i^{v_0} * (1 - \gamma^i) \tag{4.15}$$

and

$$E^i(0) = 0 \tag{4.16}$$

Next, we discuss the energy flow of the network at the iteration t_1, from time $t_1 - 1$ to time t_1 ($t_1 \geqslant 1$). We use $\Delta E^i(t_1)$ to stand for the total contained energy change of vertex i from time $t_1 - 1$ to t_1, then it can be written as:

$$\Delta E^i(t_1) = A^i(t_1 - 1) \tag{4.17}$$

The received energy for vertex i is

$$A^i(t_1) + R^i(t_1) = \sum_{k=0, k\neq i}^{N-1} R_i^k(t_1 - 1) \tag{4.18}$$

where $R_i^k(t_1 - 1)$ stands for the energy reflected from vertex k and received by vertex i. R_i^k is calculated by EDF with connection strength matrix $T' = T_{-v_0, -v_0}$. We use β_i^k to indicate $\frac{T'_{ki}}{\sum_{j=0, j\neq i}^{N-1} T'_{ji}}$, equation 4.4 can be rewritten as:

$$R_i^k(t_1) = R^k(t_1 - 1) * \beta_i^k \tag{4.19}$$

By remark 1 we have:

$$\begin{aligned} R^k(t_1) &= \sum_{j=0}^{N-1} R_j^k(t_1 - 1) * \gamma^i \\ &= \sum_{j=0}^{N-1} R^k(t_1 - 1) * \beta_j^k * \gamma^i \end{aligned} \tag{4.20}$$

where

$$\sum_{j=0}^{N-1} \beta_j^k = 1 \tag{4.21}$$

and

$$\begin{aligned} A^k(t_1) &= \sum_{j=0}^{N-1} R_j^k(t_1 - 1) * (1 - \gamma^i) \\ &= \sum_{j=0}^{N-1} R^k(t_1 - 1) * \beta_j^k * (1 - \gamma^i) \end{aligned} \tag{4.22}$$

Finally, we show varieties of dynamics of energy flow in the TEA. Figure 4.7 and Figure 4.8 show two examples, where the horizon axis represents the time step and the vertical axis represents the total contained energy in vertex i. Figure 4.7 exhibits the first two time steps, including initial step. Vertex i is not start vertex, thus it contains null energy at time $t = 0$. Due to the fast action of start vertex, the total contained energy of vertex i at time $t = \epsilon_1$ is increased to $E^i(\epsilon_1)$. After initiation, from time $t = \epsilon_1$ to time $t = 1$, the vertex i absorbs part of its received energy by slow action. These two actions make the total contained energy in vertex i decreased to $E^i(1)$ by $R^i(0)$. The same thing happened in the time period from time period $t = 1$ to time period $t = 2$.

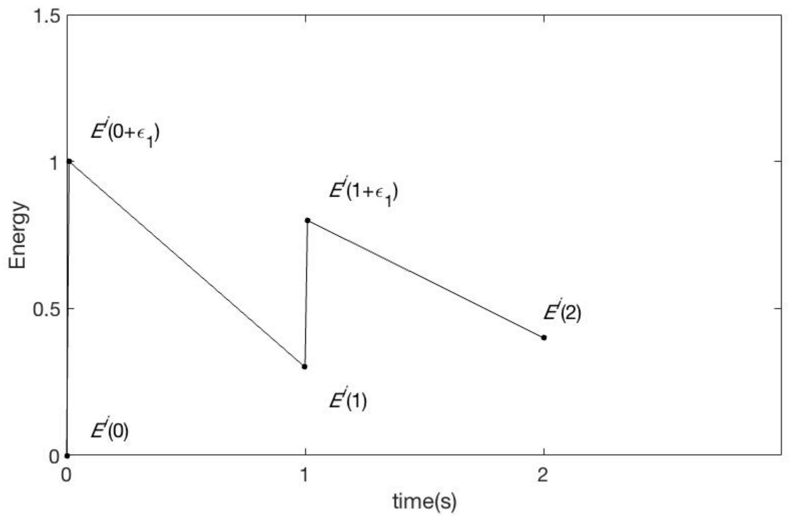
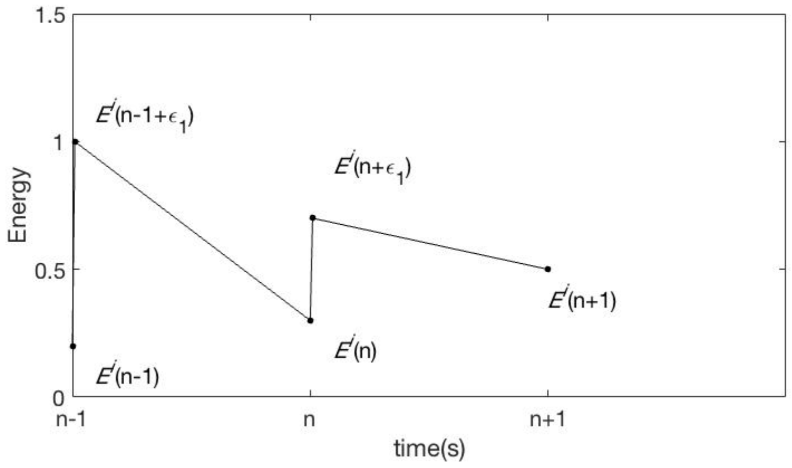


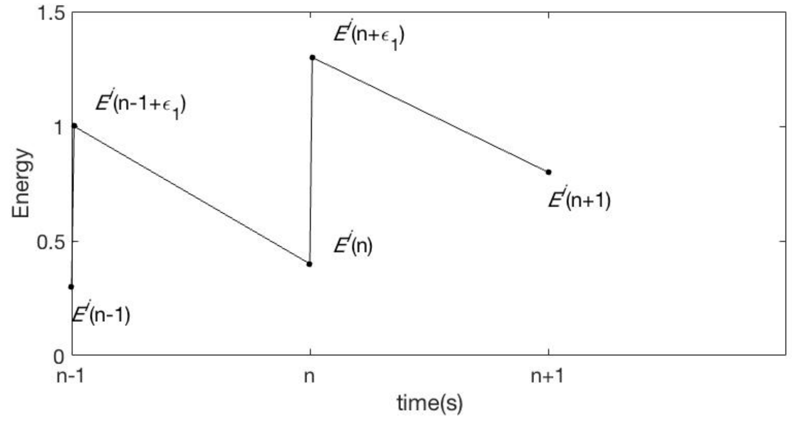
Figure 4.7: Trajectories of total contained energy evolution in the first 2 time steps

.

In Figure 4.8, we illustrated two different trajectories of evolutions of total contained energy of vertex i from $t = n - 1$ to $t = n + 1$. In (a) $E^i(n-1) > E^i(n)$ while in (b) $E^i(n-1) < E^i(n)$. We use those two figures to show the total contained energy at time $n - 1$ can be greater or smaller or equal to that at time n. This can cause the chaotic behaviors of dynamics of energy flow in TEA.



(a)



(b)

Figure 4.8: Different trajectories of energy flow dynamics

.

4.4 Summary

This chapter describes how TEA simulates the evolution of energy flow. First, we divide the energy flow process of each vertex into fast action and slow action. Second, we implement correlated rules for these two action in order to create evolution of energy flow. Third, we define the model of TEA. At last, we use examples to show that dynamics of energy flow in TEA exhibit different chaotic behaviors.

We will objectively study the energy flow in the TEA by simulation in a computational system. In the next chapter, we will introduce the implementation of a TEA based computational system.

Chapter 5

Design and Implementation of Experimental System

In the previous chapter, we introduce how TEA is designed under governing rules. Since TEA is focused on analyzing the dynamics of energy on the graph, we need to design an experimental system which is capable of helping people understand the global dynamical properties of the graph. Designing a system that can simulate TEA and reveal its properties is difficult. Besides the implementation problem of governing rules, there is a need for designing an expandable format for managing the geometry and value configuration of the graph and the property of vertices in the graph. Moreover, we need a visualization approach for displaying energy distribution dynamics of each vertex on the graph. In this section, we will introduce the experimental system for TEA-based research.

5.1 System Overview

TEA experimental system is concentrated on providing time-based energy propagation visualization services to the user. The system consists of three major modules: simulation environment controller, energy flow simulator and visualization, as Figure

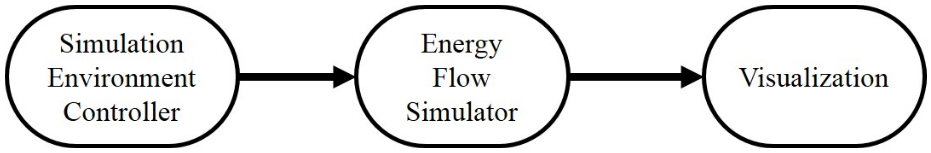


Figure 5.1: Software system architecture.

5.1 shows. The simulation environment controller module will generate the TEA network case file; the energy flow simulator module will implement the governing rules of the TEA and simulate energy flow among vertices at each time step; the visualization module will visualize the structure of the TEA network and display configuration of TEA on the graph.

We design TEA simulation system in this three-modules pattern because we want to create an application that separates the different aspects of the application (data preparation logic, data processing logic and visualization logic), while providing a loose coupling structure between these modules. The pattern specifies where each kind of logic should be located in one module of the application: data preparation logic belongs to the simulation environment controller module; data processing logic belongs in the energy flow simulator module; visualization logic belongs to the visualization module. This separation enables us to focus on one aspect of the implement at a time, which would help us manage complexity in building the application.

The loose coupling between the three main modules of TEA simulation system also reduces the risk that a change made within one module will create unexpected changes in other modules. For example, if we change the approach of calculating the connection strength matrix in simulation environment controller module while keep the same TEA network case file format, we do not need to worry about the changes in energy propagation simulator module.

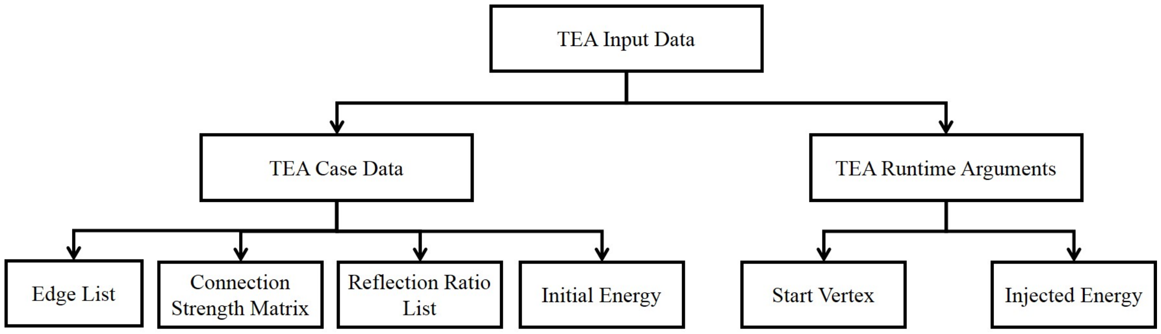


Figure 5.2: Structure of TEA simulation environment data.

5.2 Properties of System Simulation Environment

We first introduce simulation environment controller module of TEA experimental system. Figure 5.2 shows the structure of TEA input data, which consists of two components, one is TEA case data while the other is TEA runtime arguments. TEA case data includes both geometry and value configuration of graph and property of vertices. In this dissertation, we define TEA case data to be an xlsx file which provides the detailed information of edge list, connection strength matrix, reflection ratio list and initial energy. TEA runtime arguments, which include start vertex and injected energy, will be inputed at the initial state of TEA system in the energy flow simulation module. The system simulation environment is mainly about generating TEA case data.

In this section, we will introduce algorithm of generating connection strength matrix in subsection 5.2.1; after that, we will introduce the format of TEA case data in subsection 5.2.2. In this part, we assume the name of each vertex in N-vertex TEA graph is labeled from 0 to $N - 1$.

5.2.1 Algorithm of Generating Related Parameters

We firstly illustrate the complete steps of generating connection strength matrix. The sketch of this algorithm is shown in Algorithm 1. We start from acquiring the list of vertices n in the edge list l(line 2); connection strength matrix is initialized as a $N \times N$ matrix E (line 3); for each pair of vertices in l, algorithm will measure the shortest distance between them, and fill the corresponding element of matrix E with connection strength which is calculated by the measured shortest distance (lines 4-9).

Algorithm 1 Algorithm of Generating Connection Strength Matrix

```
1:  procedure GetEMatrix($l$)                                  ▷ $l$ is edge list
2:      $n$ = GetVerticesList($l$)                              ▷ ($n$ is list of vertices in $l$)
3:      $E$ = Init($l$)                                         ▷ $E$ is connection strength matrix of $l$
4:      for $i$ = 0,1,2,...,sizeof($n$) do
5:          for $j$ = 0,1,2,...,sizeof($n$) and $j$ != $i$ do
6:              $k$ = FindShortestPath($l, n[i], n[j]$)
7:              $E[n[i], n[j]]$ = 1/power($2, k$)
8:          end for
9:      end for
10:     return $E$
11: end procedure
```

Multiple metrics can be used to measure shortest distance between each pair of vertices in the edge list. In this subsection, an example of measuring geodesic distance [39] is implemented as Algorithm 2 shows. This approach is based on non-recursive Breadth-First Search (BFS) method. As we assume TEA network is a connected graph, the function will return the geodesic distance in line 10. Otherwise, we do not need to consider the situation that the function returns in line 20 since we do not discuss the disconnected TEA network in this dissertation.

Algorithm 2 Approach of measuring shortest path between two vertices.

```
procedure FindShortestPath($l$,$i$,$j$)    ▷ $l$ is edge list; $i$ is from vertex; $j$ is to vertex
    $d = 1$    ▷ $d$ is measurement of shortest distance
    create vertex set $Q$    ▷ $Q$ contains all vertices in edge list.
    $Q$.remove($i$) create set $P$ and $P$.push($i$)
    while $P$ is not null do
        create set $O$
        for each element $u$ in $Q$ do
            for each neighbor $v$ of $u$ do
                if $v == j$ then
                    return $d$
                else if $v$.contains($Q$) is not true then
                    $O$.add($v$)
                    $Q$.add($v$)
                end if
            end for
        end for
        $d = d + 1$
        $P = O$
    end while
    return $INFINITY$
end procedure
```

5.2.2 Case Definition and Data Format for Experiments

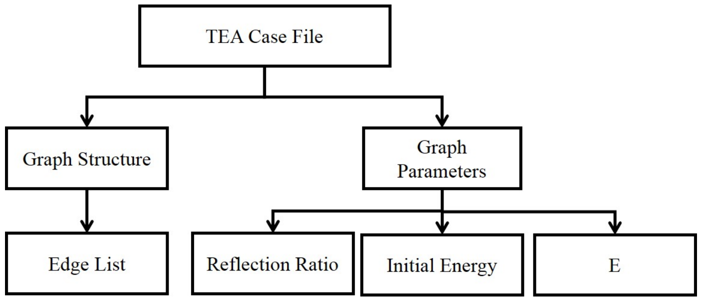


Figure 5.3: Structure of TEA case data.

The simulation case file includes all the basic information of TEA case file to simulate the energy flow on the graph. This file consists of two parts as Figure 5.3 shows. Graph structure data contains the edge list data of the graph, which is a $M \times 3$ matrix where M is the number of edges in the graph; graph parameter data consists of three matrices, including two $1 \times N$ row vectors and one $N \times N$ matrix where N is the number of vertices. Two row vectors include reflection ratio vector and initial energy vector of TEA, while the $N \times N$ matrix is the connection strength matrix of TEA which we labeled as E in our program. In this software, we use xlsx format for storing and transferring simulation case data because this format is easy to storing, organizing and manipulating by Microsoft Excel.

Edge Number	From Vertex No.	To Vertex No.
1	1	2
2	2	3
3	1	4
4	2	0
5	3	5
6	4	0
7	0	5
$\vdots$	$\vdots$	$\vdots$
M	v_i	v_j

Table 5.1: Example of TEA case file graph structure.

Reflection Ratio								
0.9	0.3	0.5	0.2	0.7	0.3	$\cdots$	0.3	0.1
Initial Energy								
0	0	0	0	0	0	$\cdots$	0	0
E								
0	0.5	1	0.5	1	1	$\cdots$	1	0.5
0.5	0	1	0.5	1	0.25	$\cdots$	0.25	0.125
1	1	0	1	0.5	0.5	$\cdots$	0.5	0.25
0.5	0.5	1	0	0.25	1	$\cdots$	0.25	0.5
1	1	0.5	0.25	0	0.5	$\cdots$	0.5	0.25
1	0.25	0.5	1	0.5	0	$\cdots$	0.5	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
1	0.25	0.5	0.25	0.5	0.5	$\cdots$	0	1
0.5	0.125	0.25	0.5	0.25	1	$\cdots$	1	0

Table 5.2: Example of TEA case file graph parameters.

Figure 5.1 and Figure 5.2 show examples of TEA network case data in N-vertex and M-connection system. In Figure 5.1, each row stands for a connection, i.e, row five is the fifth connection of the network, and this connection is between vertex three and vertex five. In Figure 5.2, row two is the reflection ratio vector of the TEA network, i.e, the data in second cell of reflection ratio vector is the reflection ratio for the vertex one (we use zero-based numbering); row four is the initial energy vector of the TEA network, in this example the initial energy for each vertex is set as null energy; the matrix starting from row six is the $N \times N$ matrix of connection strength matrix, i.e, the element in row three and column four of this matrix is the connection strength between vertex two and vertex three.

5.3 Software Implementation

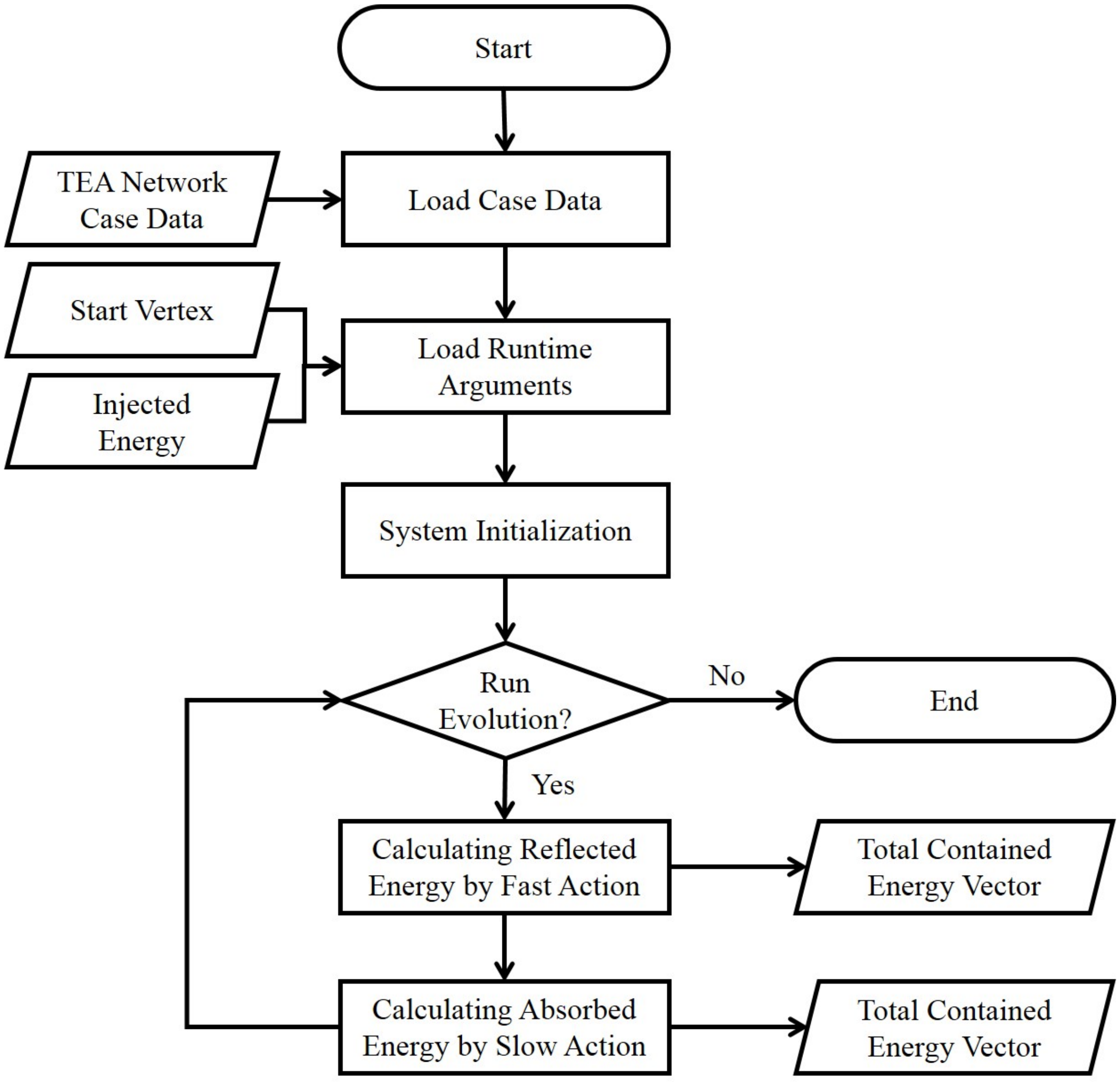

Start
TEA Network Case Data
Load Case Data
Start Vertex
Injected Energy
Load Runtime Arguments
System Initialization
Run Evolution?
No
End
Yes
Calculating Reflected Energy by Fast Action
Total Contained Energy Vector
Calculating Absorbed Energy by Slow Action
Total Contained Energy Vector

Figure 5.4: Flow chart of energy flow simulator.

The implement of energy flow simulator module follows the operations of TEA as we introduced in the previous chapter, Figure 5.4 shows the flow chart of the evolution simulator:

- Load Case Data: As we introduced in the previous section, TEA network case data is stored in xlsx format. As a result, most of the parsing process need to be done in this step;

- Load Runtime Arguments: We read the inputs of TEA runtime arguments, including start vertex and injected energy;
- System Initialization: In this step, we implement the equation 4.14 of the previous chapter. We also initialize absorbed energy vector A, reflected energy vector R and total contained energy vector E as TEA model defined. Assume N is the number of vertices in the network: absorbed energy vector A is a $1 \times N$ row vector which is initialized by equation 4.14; reflected energy vector R is $1 \times N$ zero row vector; and total energy vector E is a $1 \times N$ row vector which is initialized by initial energy vector in the TEA network case file;
- Calculating Reflected Energy by Fast Action: In this step, we implement the equation 4.22 from the previous chapter. After that, we update contained total contained energy vector E by

  $$E_i = E_i + A_i + R_i \tag{5.1}$$

  where E_i and A_i stand for the ith element of E and A. The result of E will output to the visualization module;
- Calculating Absorbed Energy by Slow Action: In this step, we implement the equation 4.20 from the previous chapter. After that, we update total contained energy vector E by

  $$E_i = E_i - R_i \tag{5.2}$$

  where E_i and R_i stand for the ith element of E and R. The result of E will output to the visualization module.

The design of energy flow simulator module is highly modular and thus flexible. Due to the modularity design, it is extremely easy for us to add or modify models in order to expand the capabilities of TEA simulation system. This feature eliminates the need to make major changes on the main body of the program, when we need to modify or integrate new models.

5.4 Method for Result Visualization

The visualization module is focused on helping people understand and analyze the energy flow in the TEA network by visualization techniques. TEA simulation system visualization plan includes TEA network visualization and displays of time-based TEA configuration.

The basic visualization problem can be simply expressed as follows: given a graph formed by nodes and connections, calculate the position of each node and draw the corresponding curve for every connection[23]. Thus, one of the key factors of visualization is how to lay all the vertices and connections on a plane. Indeed, a good layout allows a user to extract information at a glance, whereas a poor layout is far more challenging to be understood.

As we mentioned in the Chapter 2, force-directed method is used to generate the layout of TEA graph [25]. Figure 5.5 shows the layout generated by force-directed method of a forty-nine vertex lattice graph.

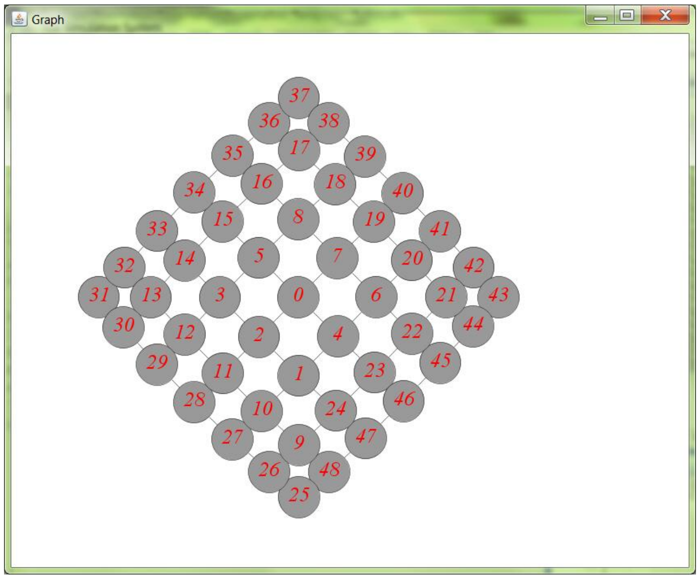
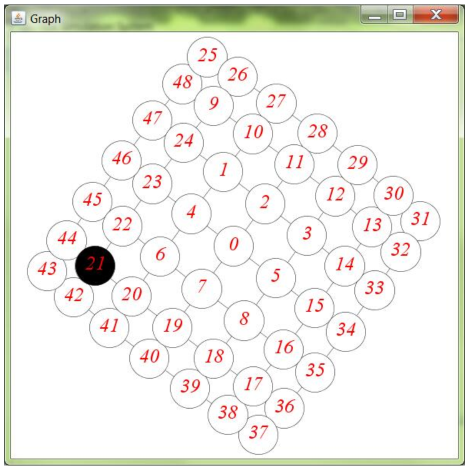
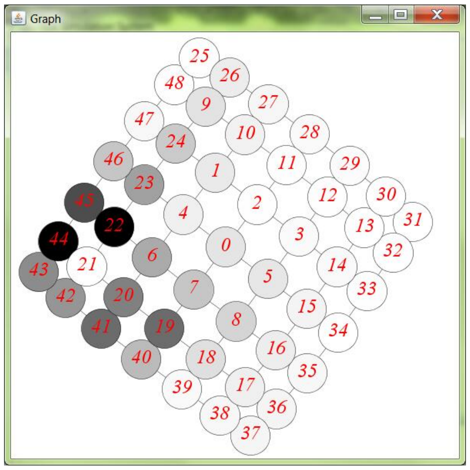


Figure 5.5: The force-directed layout of forty-nine vertex lattice.

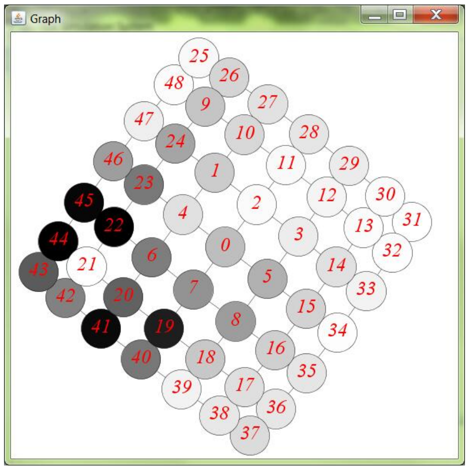
TEA experimental system defines the gray scale of each node represents partial energy information of corresponding vertex in the TEA graph. Darker color represents more energy is distributed on the correlated vertex at current time step, while lighter color represents less energy is distributed on the correlated vertex at current time step. Figure 5.6 shows an example of visualized configuration of TEA in the first five time step. This TEA is composed of forty-nine vertices with the start vertex twenty-one and injected energy forty. The characteristic manner of this example is clear: at initial step $t = 0$, the energy was injected to start vertex twenty-one; from time step $t = 1$ to $t = 3$, the injected energy reflected from start vertex and distributed over the network; start from time step $t = 4$, energy distribution remains stable in the network.



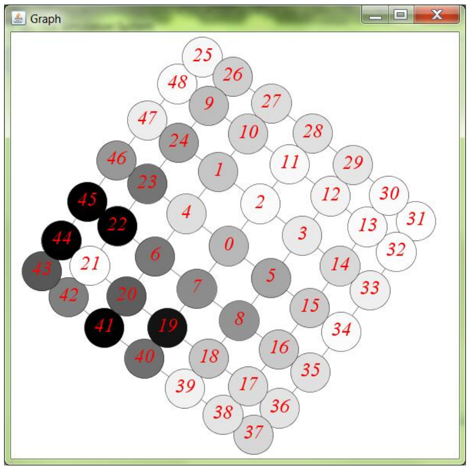
(a) t=0



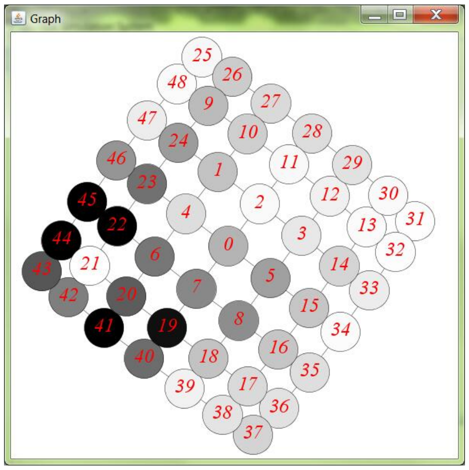
(b) t=1



(c) t=2



(d) t=3



(e) t=4

Figure 5.6: First five time steps of the forty-nine vertex graph TEA with start vertex six and injected energy forty.

The front-end graphical user interface (GUI) is developed by Java as Figure 5.7 shows: this GUI consists of nine components, labeled from one to nine. In the rest of this section, all these components are described in the ascending order.

- Component one is the control panel. This panel provides access to the following functions: button Load stands for loading a TEA network case data; button Visualization stands for visualizing TEA network; button Initialization stands for reading TEA runtime arguments and initializing TEA system; button Step stands for processing TEA by one time step; and button Ten Steps stands for processing TEA by ten time steps.

- Component two is implemented for inputing TEA runtime arguments.

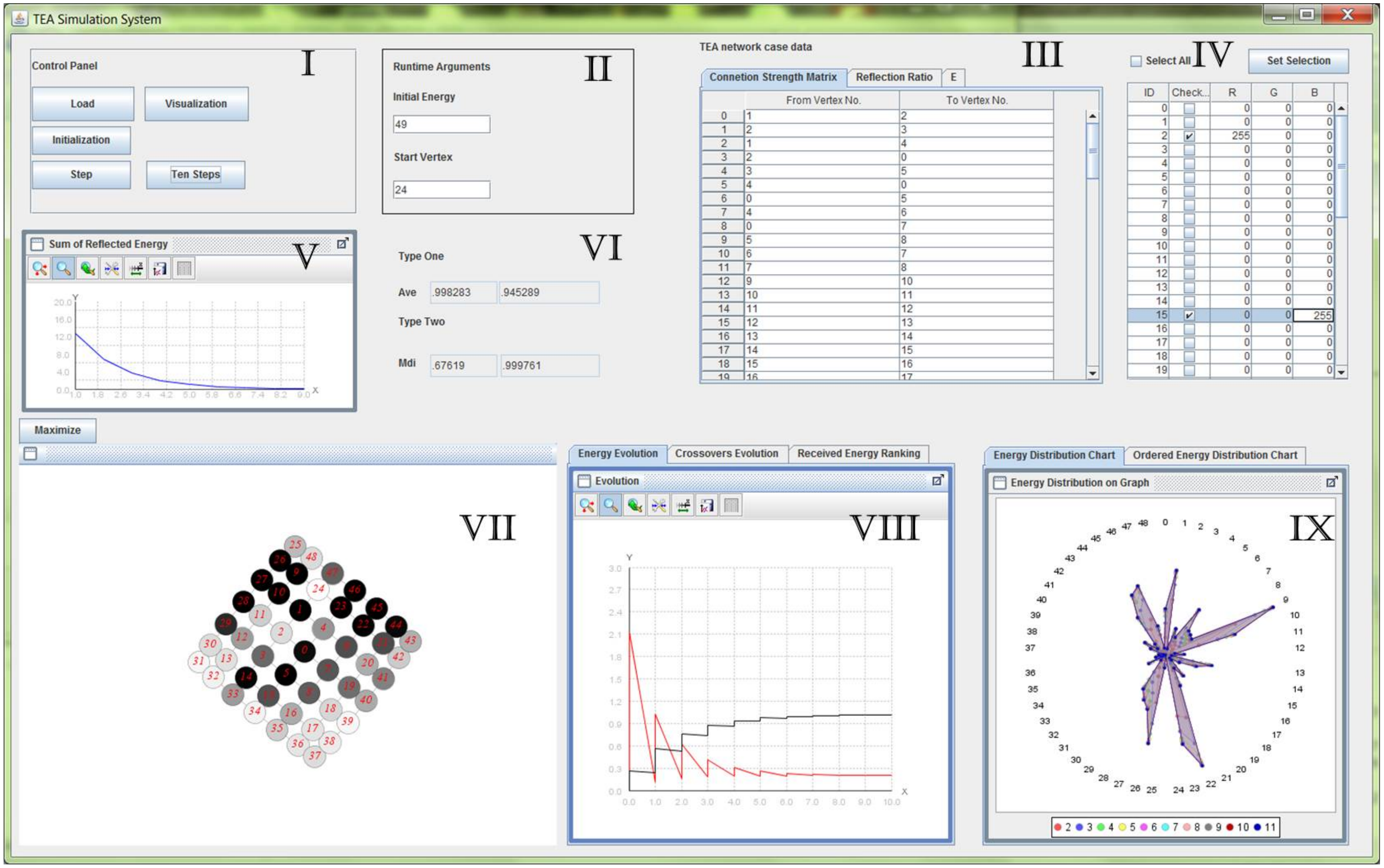
- Component three is implemented for displaying and modifying TEA network case data.

- Component four is implemented for selecting vertices and setting colors of the selected vertices. The energy dynamics of selected vertices will be visualized in component eight.

- Component five is implemented for displaying the evolution of sum of entire reflected energy in TEA.

- Component six is implemented for illustrating the runtime statistic of TEA at each time step. Type one shows the average absorbed energy among the TEA system and the standard deviation of this mean; type two shows the median of the absorbed energy among the TEA system and the standard deviation of this median.

- Component seven is implemented for displaying the absorbed energy distribution of TEA network at each time step.



Runtime Arguments

Initial Energy

49

Start Vertex

24

Type One

Ave .998283 .945289

Type Two

Mdi .67619 .999761

TEA network case data

Connetion Strength Matrix | Reflection Ratio | E

	From Vertex No.	To Vertex No.
0	1	2
1	2	3
2	1	4
3	2	0
4	3	5
5	4	0
6	0	5
7	4	6
8	0	7
9	5	8
10	6	7
11	7	8
12	9	10
13	10	11
14	11	12
15	12	13
16	13	14
17	14	15
18	15	16
19	16	17

Select All

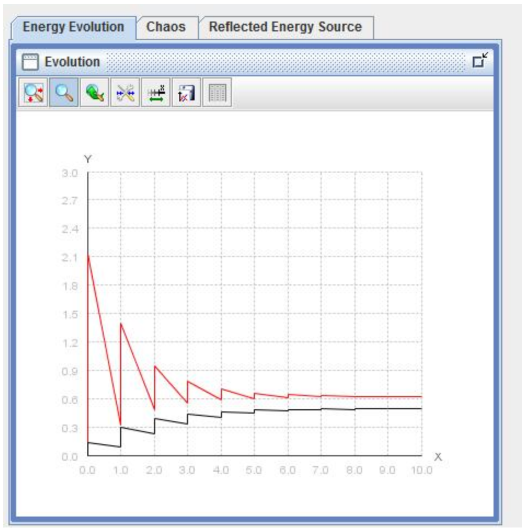
Set Selection

ID	Check...	R	G	B
0	☐	0	0	0
1	☐	0	0	0
2	✓	255	0	0
3	☐	0	0	0
4	☐	0	0	0
5	☐	0	0	0
6	☐	0	0	0
7	☐	0	0	0
8	☐	0	0	0
9	☐	0	0	0
10	☐	0	0	0
11	☐	0	0	0
12	☐	0	0	0
13	☐	0	0	0
14	☐	0	0	0
15	✓	0	0	255
16	☐	0	0	0
17	☐	0	0	0
18	☐	0	0	0
19	☐	0	0	0

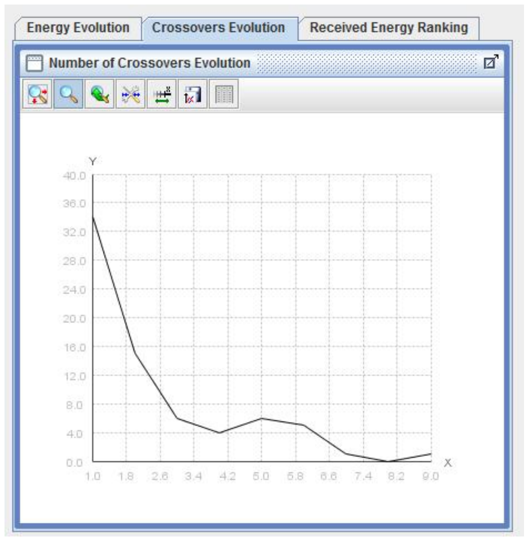
Figure 5.7: The GUI of TEA simulation system.

- Component eight contains three sub-components. Figure 5.8a shows the energy evolution of selected vertices. In this figure horizon axis represents the time step while the vertical axis represents the energy distributed on the correlated vertex at the related time step. The vertices and its correlated color can be selected in component four. Figure 5.8b shows the irregularity in energy dynamics. We use crossovers among total contained energy evolution to represent the irregularity in energy dynamics of the graph. In this figure, the horizon axis represents the time step while the vertical axis represents the number of crossovers at correlated time step. Figure 5.8c shows the received energy ranks of the selected vertex. In this figure, the horizon axis represents the name of vertex while the vertical axis represents the amount of energy received from correlated vertex to selected vertex at current time.

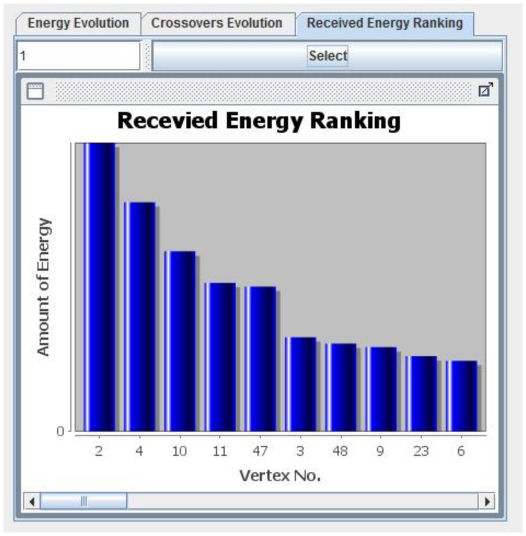
- Component nine displays two kinds of energy distribution charts at each time step. In Figure 5.9a and Figure 5.9b, the length of the spoke stands for the entire energy absorbed by the correlated vertex. Figure 5.9a shows charts of vertices energy distribution at $t = 2$ and $t = 3$ where the ith spoke represents the vertex i. Figure 5.9b shows charts of ordered energy distribution at $t = 2$ and $t = 3$ where the ith spoke represents the vertex which absorbs ith least entire energy. In those two Figures, different colors of charts stand for different time step: the charts in red line stand for energy distribution or ordered energy distribution at time step $t = 2$; the charts in blue line stand for energy distribution or ordered energy distribution at time step $t = 3$.



(a) Energy distribution dynamics.

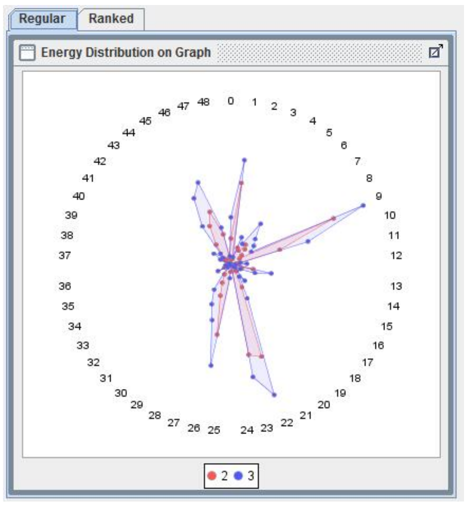


(b) Irregularity in energy dynamics.

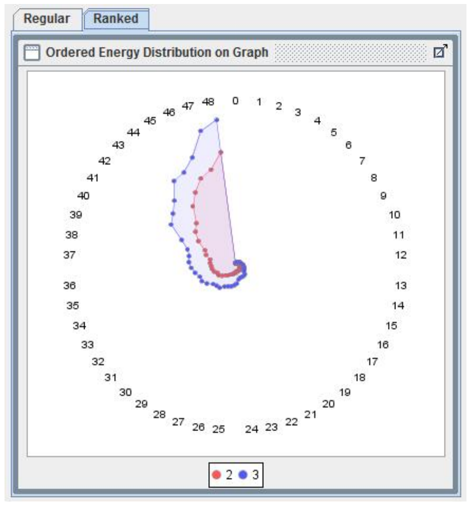


(c) Received energy ranks for the selected vertex.

Figure 5.8: TEA real time energy evolution monitors.



(a) Energy distribution chart.



(b) Ranked energy distribution chart.

Figure 5.9: Two kinds of TEA real time energy distribution charts.

5.5 Summary

In this chapter, we introduce our TEA experimental system in detail. This system consists of three modules: simulation environment controller generates the TEA network case data; energy flow simulator reads the generated case data, accepts runtime arguments and simulate the dynamics of energy flow; visualization module displays the time-based energy propagation. In this dissertation, we mainly use Matlab and Java to develop this system; we use JAMA, which is a linear algebra package in Java, to do the matrix calculation; since TEA network case data is in excel format, we use Apache POI components, which are open source and third party API for developing pure Java ports of xlsx format, to parse case data; and in data visualization module we choose Gephi-toolkit, which is a Java base library, to layout the graph.

In the next chapter, we will discuss the impacts of different factors on the evolution of energy distribution in the TEA.

Chapter 6

Impacting Factors on Energy Distribution

TEA consists of several different arguments, including reflection ratio for each vertex, the location of energy injection, network structure, and structure of reflection ratio distribution. Each of these arguments generates a different evolution of energy distribution on the graph. These results will help people understand the impacts of different factors on dynamics of energy distribution.

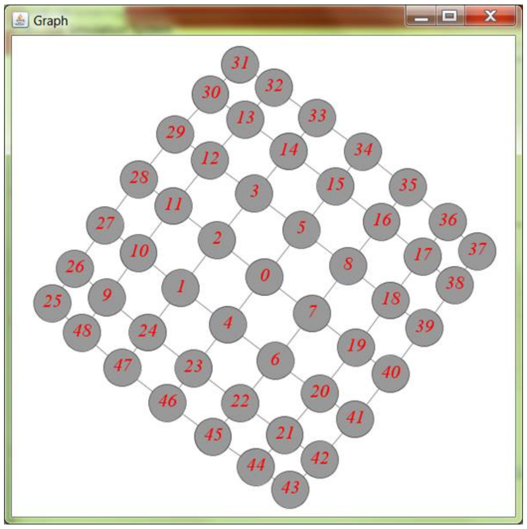
In this chapter, we design four different kinds of experiments. Each experiment will contain several simulations that will separate one kind of impacting factor, as well as analyze the impacts of the factor. Impacts include energy distribution dynamics, irregularity of dynamic evolution, the homogeneous or heterogeneous feature of energy distribution, and the convergence speeds of energy evolution. Moreover, a conclusion is presented by comparing the results of energy distribution among these impacting factors.

6.1 Impacts of Homogeneously Reflection Ratios Distribution

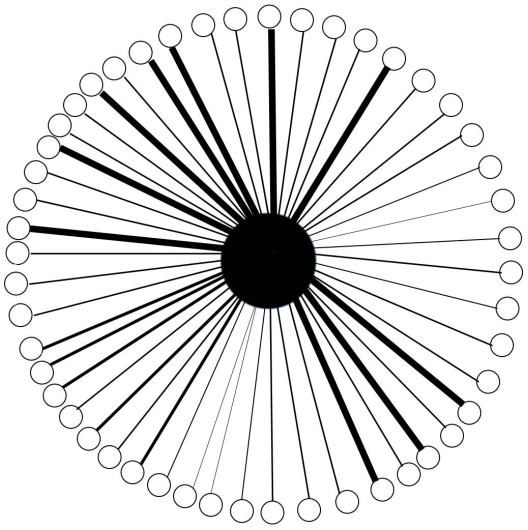
We start by discussing the impacts of homogeneously distributed reflection ratios on the evolution of energy distribution in the graph. Reflection ratio stands for the ability of a vertex to reflect its received energy; homogeneously distributed stands for the reflection ratio of each vertex is uniform. When the reflection ratio of each vertex on the graph is defined, we become interested in how energy is distributed on the graph, what are the energy distribution dynamics and when did these dynamics converge.

In order to explain these questions, we design an experiment of paired comparison simulations. These two simulations are based on forty-nine vertex homogeneous lattice graph, which is shown in Figure 6.1a. In this square grid graph, we assume that each connection has the same distance of one. By the Algorithm 1 in section 5.2, we can calculate the connection strength matrix of the graph. Moreover, in order to compare the impacts of different homogeneously distributed reflection ratio on the graph, we will adjust the value of reflection ratio vector. The reflection ratio vector is shown in the Table 5.2 row two, which is in the TEA network case data. Figure 6.1b shows the energy distribution among the vertices of the graph: the energy disperses from being concentrated in the center black vertex to becoming spread out, and later it gets distribute on the other forty-eight white vertices. The weight of each line in graph stands for the amount of energy transfer from center black vertex to the correlated vertex.

This section consists of three parts: subsection 6.1.1 discusses impacts of homogeneously distributed low reflection ratios on the graph; subsection 6.1.2 discusses impacts of homogeneously distributed high reflection ratios; subsection 6.1.3 gives a summary of impacts of homogeneously distributed reflection ratios.



(a) A forty-nine vertex homogeneous lattice graph with the length of each connection is one.



(b) Energy disperses from the center black vertex to the other forty-eight white vertices. The weight for each line stands for amount of energy transfer from black vertex to the correlated vertex.

Figure 6.1: Model of the simulation.

6.1.1 Low Reflection Ratios

In this part, we will present the impacts of homogeneously distributed low reflection ratios by a simulation. Low reflection ratio of a vertex can be explain as, with certain amount of received energy, most of them are absorbed by the vertex while only a small part of them will be reflected. Figure 6.2 shows the received energy model of vertex with low reflection ratio. In this simulation, we set the reflection ratio of every vertex to be 0.1 and set the runtime arguments as start vertex to be 0 (the center of lattice graph) and injected energy to be 5.

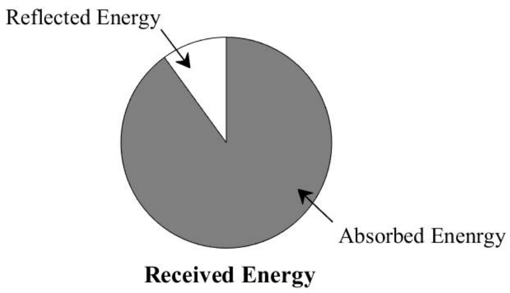
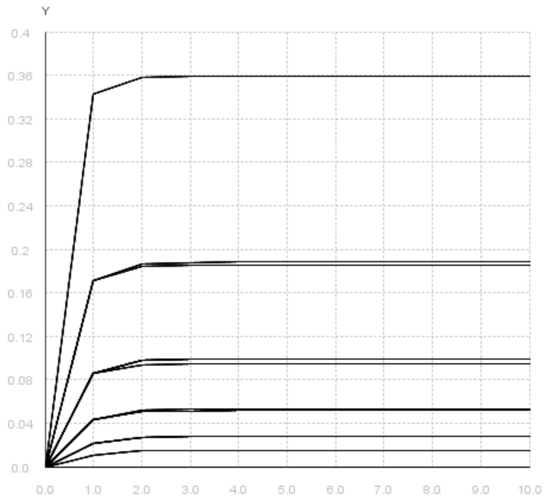


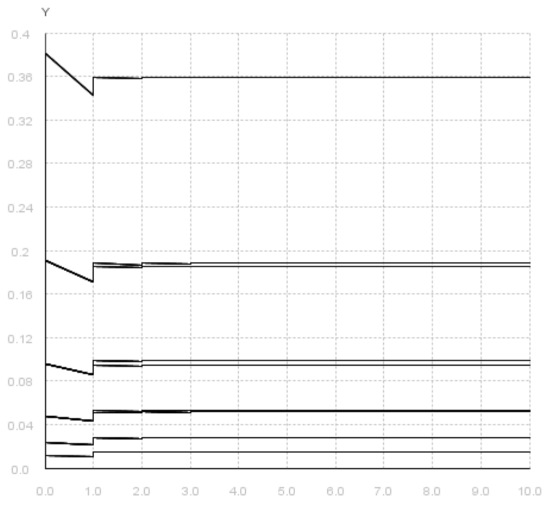
Figure 6.2: Model of low reflection ratio vertex.

Figure 6.3 shows the results of energy distribution dynamics in the whole process. It can be seen that the homogeneously distributed low reflection ratios significantly decrease the entire energy flow in the graph. Figure 6.3a shows the entire absorbed energy dynamics converge very quickly in this simulation. Figure 6.3b shows total contained energy dynamics also converge rapidly. This is because for each vertex, low reflection ratio affects most of energy absorbed by each vertex. Thus, the activities of the whole system are decreased.

Figure 6.5 exhibits the entire absorbed energy distribution at different time steps. From the statistic component of TEA simulation system, we can see the average absorbed energy for each vertex is 0.104 and the standard deviation (SD) of absorbed energy is 0.095. Thus, we can calculate relative standard deviation (RSD) of absorbed energy which is $(0.095/0.104) * 100\% = 91.3\%$. Since the relative standard deviation is so high, we consider that the energy is distributed unevenly on the graph.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.3: Energy distribution dynamics for a grpah with low reflection ratios homogeneously distributed.

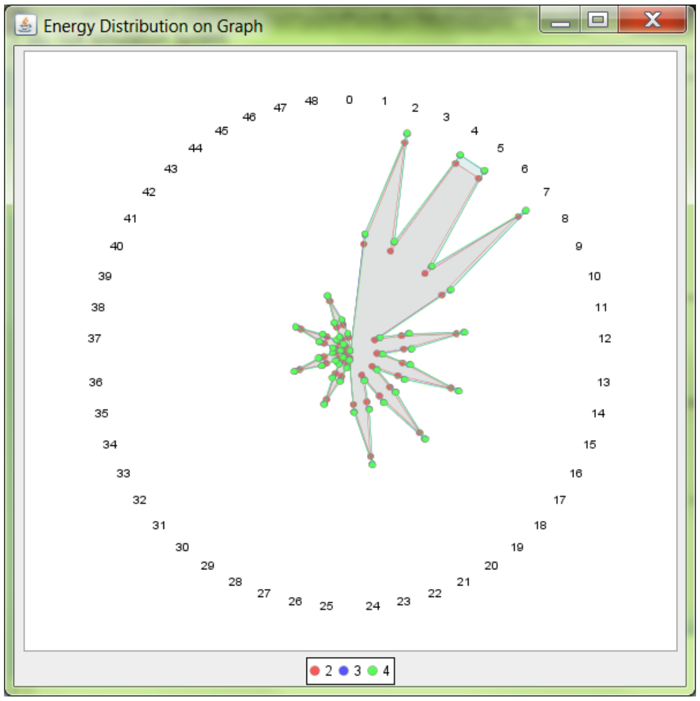
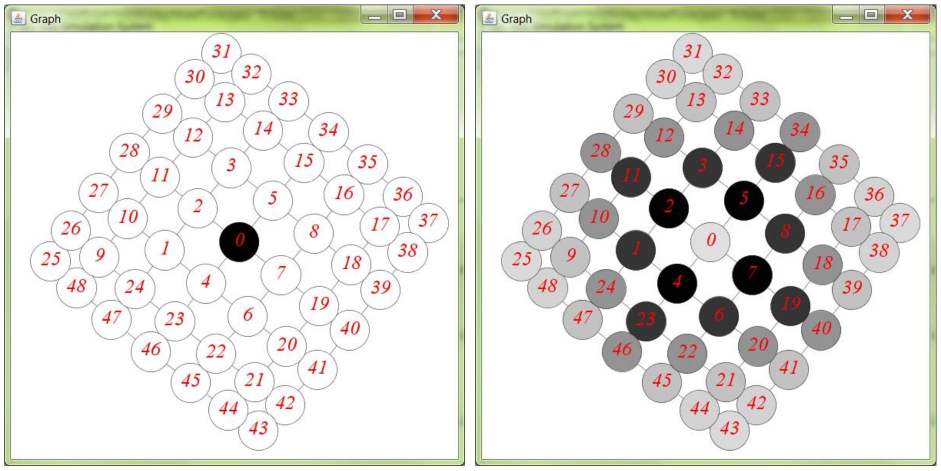


Figure 6.4: Entire absorbed energy distribution chart for a graph with low reflection ratios homogeneously distributed.



(a) t=0 (b) After t=1.

Figure 6.5: Absorbed energy distribution fo a graph with low reflection ratios homogeneously distributed.

6.1.2 High Reflection Ratios

In this part, we will show the impacts of homogeneously distributed high reflection ratios. High reflection ratio for a vertex means that with certain amount of received energy, most of them are reflected by the vertex while only a small part of them will be absorbed. Figure 6.6 shows the received energy model of each vertex with high reflection ratio. In this simulation, we assume the reflection ratio of each vertex as 0.9 and set runtime arguments as start vertex to be 0 and injected energy to be 5.

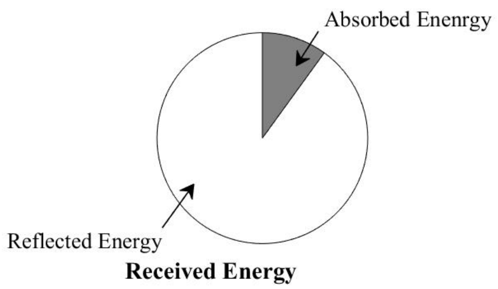
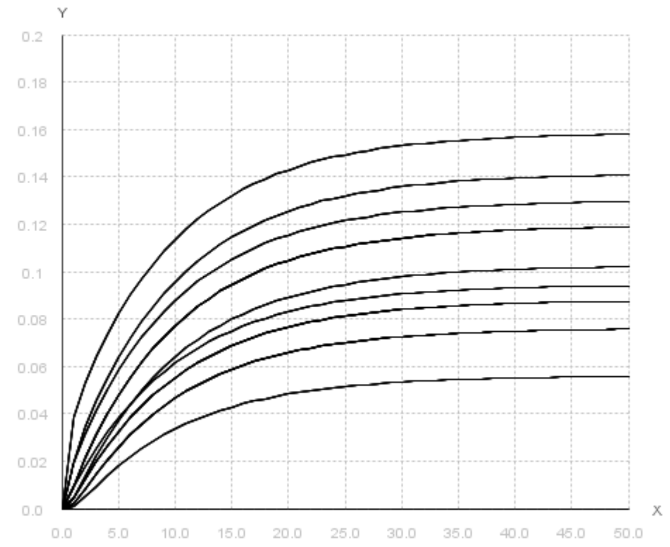
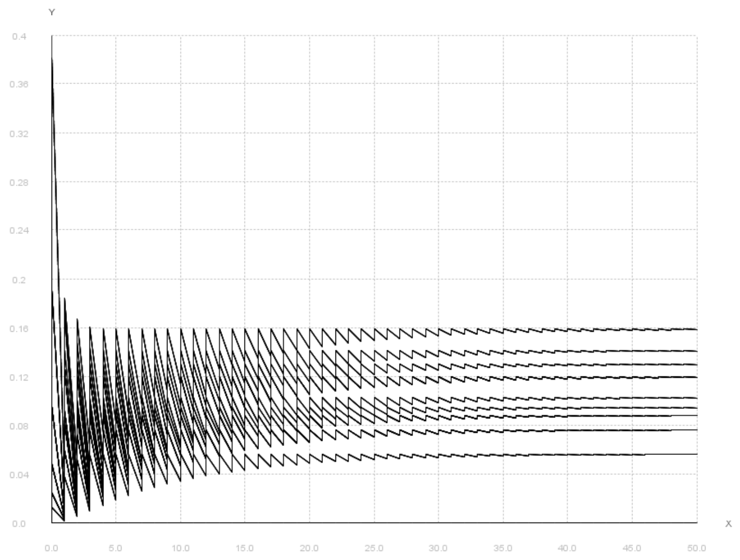


Figure 6.6: Model of high reflection ratio vertex.

Figure 6.7 shows the results of energy distribution dynamics in the whole process. It can be seen that the homogeneously distributed high reflection ratios obviously increase the energy flow in the graph. Figure 6.7a shows the entire absorbed energy dynamics converge very slowly in the simulation. Figure 6.7b shows the total contained energy dynamics exhibit chaotic oscillations. That is because for each vertex, high reflection ratio affects most of energy reflected to the other vertices. This will amplify the energy flow in the graph. Thus, the activities of the whole system are increased.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.7: Energy distribution dynamics on a graph with high reflection ratios homogeneously distributed.

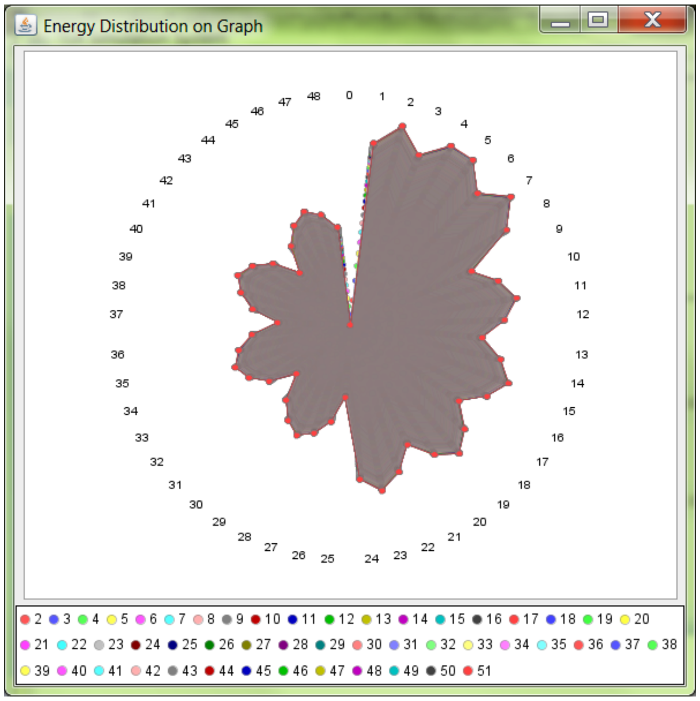
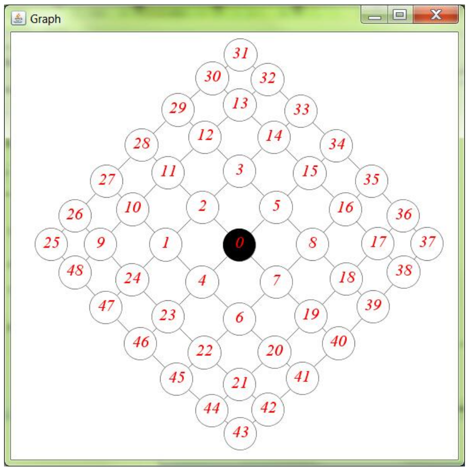


Figure 6.8: Entire absorbed energy distribution chart on a graph with high reflection ratios homogeneously distributed.

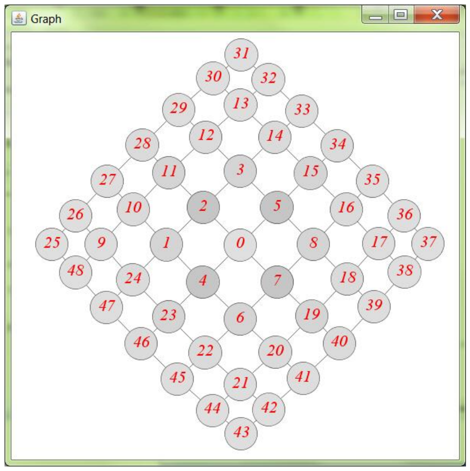
Figure 6.7 exhibits the entire absorbed energy distribution at five different steps. In this experiment, the average absorbed energy for each vertex which is 0.104 and the standard deviation of absorbed energy which is 0.029. Thus, the relative standard deviation of absorbed energy which is $(0.029/0.104) * 100\% = 27.9\%$. Since the relative standard deviation is so low, we consider the energy is distributed evenly on the graph.

6.1.3 Remarks

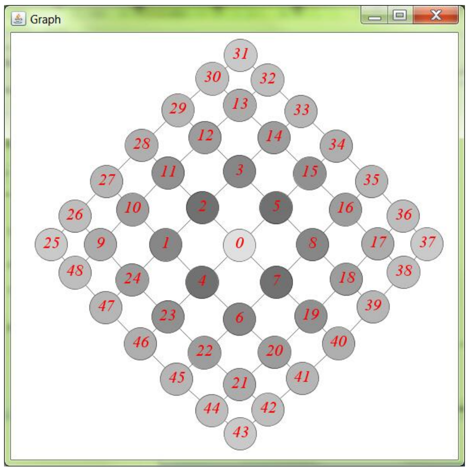
In this section, we use paired comparison simulations to illustrate the impacts of energy distribution dynamics through homogeneously distributed reflection ratios. By comparing the results between the two simulations, where one is based on forty-nine vertex homogeneous lattice graph with homogeneously distributed low reflection ratios and energy injected at graph center, while the other is based on forty-nine vertex



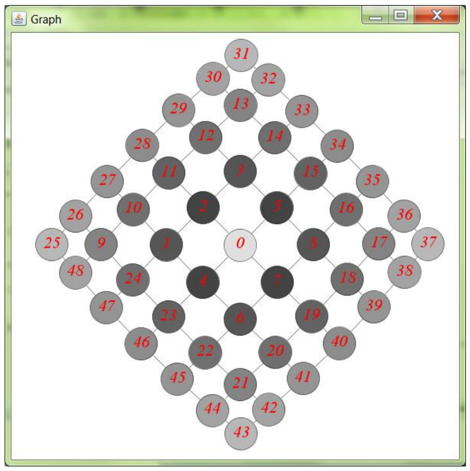
(a) t=0



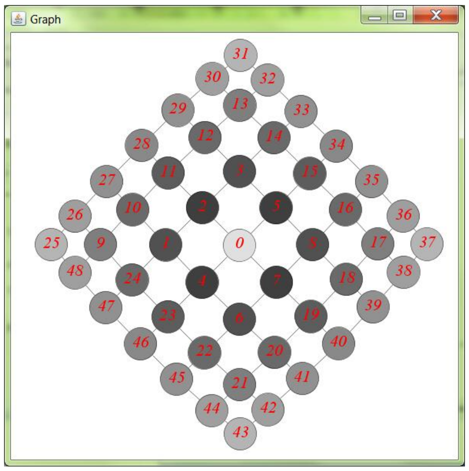
(b) t=1.



(c) t=10



(d) t=30



(e) After t=50

Figure 6.9: Absorbed energy distribution on a graph with high reflection ratios homogeneously distributed.

homogeneous lattice graph with homogeneously distributed high reflection ratios and energy injected at graph center, we can conclude:

- The higher the reflection ratio, the longer the time spend on the energy distribution dynamics converge;
- The higher the reflection ratio, the more chaotic the oscillations of energy distribution dynamics;
- The energy prefers to distribute evenly on the graph when reflection ratio is high;
- The energy prefers to distribute unevenly on the graph when reflection ratio is low;

6.2 Impacts of Different Energy Injected Location

Next, we discuss the impacts of different energy injected locations on energy distribution of the graph. Location of energy injection is presented by start vertex in the runtime arguments. With different energy injection locations, the energy distribution dynamics show variety of different patterns. When the location of energy injection is defined, we become interested in evolution of both irregularity in energy dynamics and energy distribution dynamics.

In order to demonstrate these impacts, we also design an experiment of paired comparison simulations. These two simulations are based on forty-nine vertex homogeneous lattice graph with homogeneously distributed reflection ratios. Figure 6.10 shows the received energy model for each vertex. Specifically in this section, we assume the reflection ratio for each vertex to be 0.8.

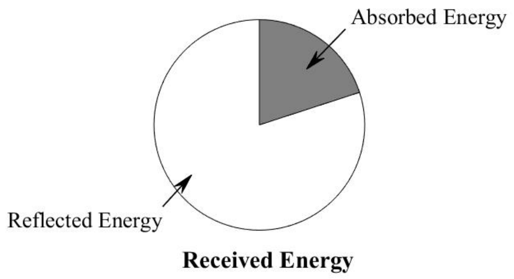


Figure 6.10: Model of vertex in simulations of different energy injected locations.

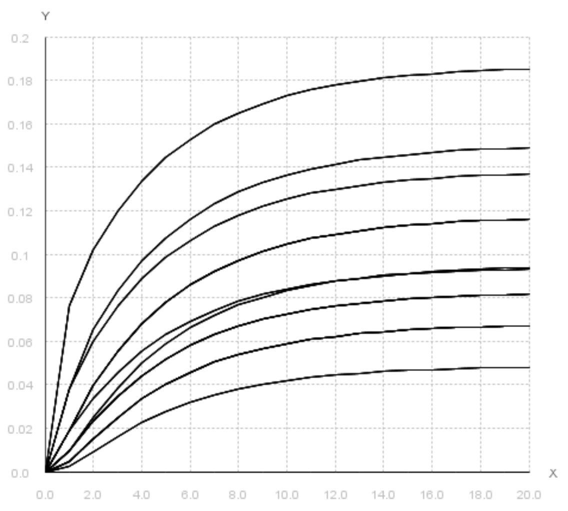
This section consists of three parts: subsection 6.2.1 exhibits the impacts of energy injected at center of graph; subsection 6.2.2 exhibits impacts of energy injected near the corner of the graph; subsection 6.2.3 give a conclusion of previous simulations.

6.2.1 Energy Injected into Graph Center

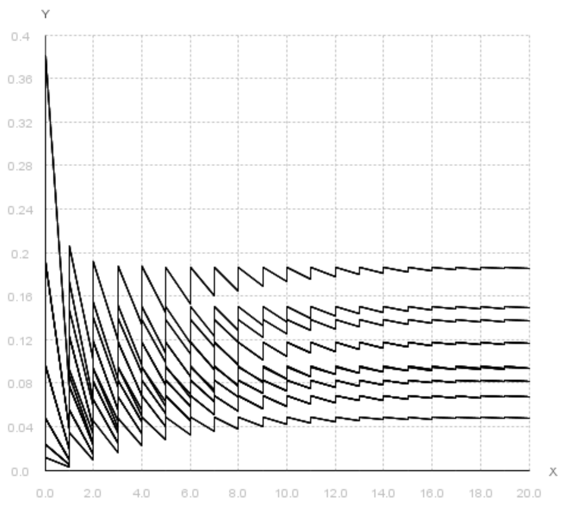
In this subsection, we will show the pattern of energy distribution dynamics when the energy is injected into the center of graph. Here, we set the runtime arguments as start vertex to be 0 and injected energy to be 5.

Figure 6.11 and Figure 6.12 show the results of the simulation. In Figure 6.11, the entire absorbed energy dynamics for forty-eight vertices exhibit nine different oscillations. Figure 6.12 shows there are less irregularities during the whole energy flow process. The reason is that the energy injected location is the center symmetry vertex of the graph. Thus, the connection strength vector for this vertex distributed homogeneously. Therefore, the received energy for each vertex at any time step will be distributed symmetrically on the graph.

Figure 6.14 shows the entire absorbed energy distribution at four different time steps. In this simulation, the average absorbed energy for each vertex is 0.103 and the standard deviation of absorbed energy is 0.038. Thus, the relative standard deviation of absorbed energy which is $(0.038/0.103)*100\% = 36.9\%$. Since the relative standard deviation is low, we consider the energy is distributed evenly on the graph.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.11: Energy distribution dynamics on a graph with energy injected from graph center.

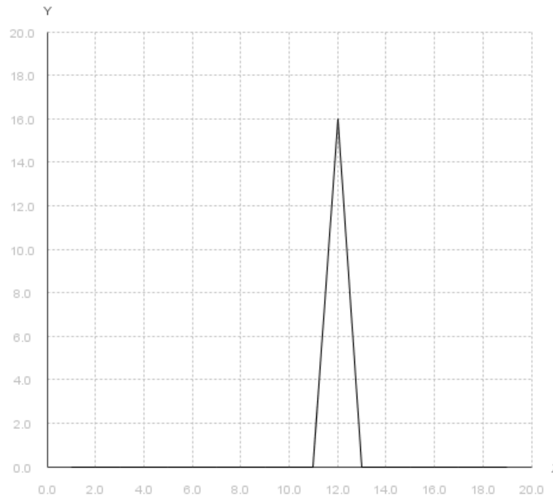


Figure 6.12: Irregularity in energy dynamics of the graph with energy injected from graph center.

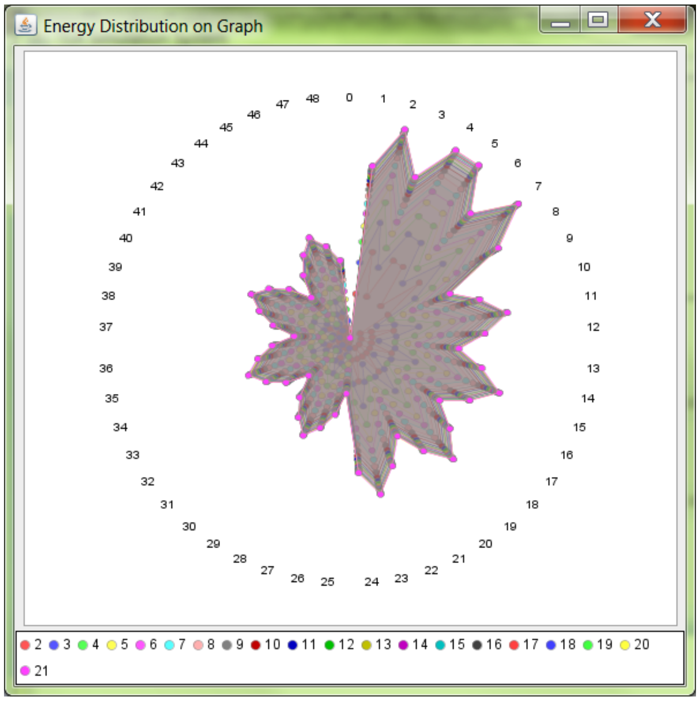
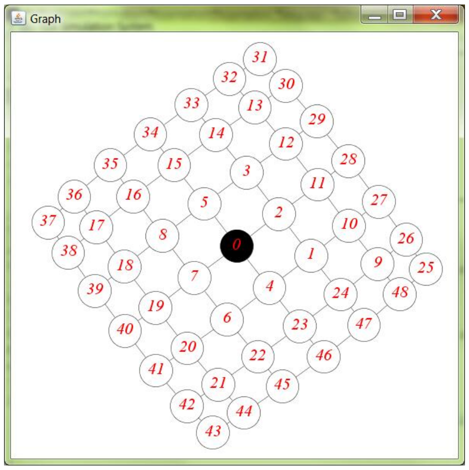
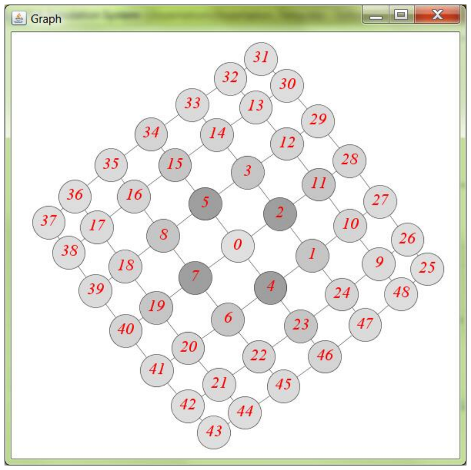
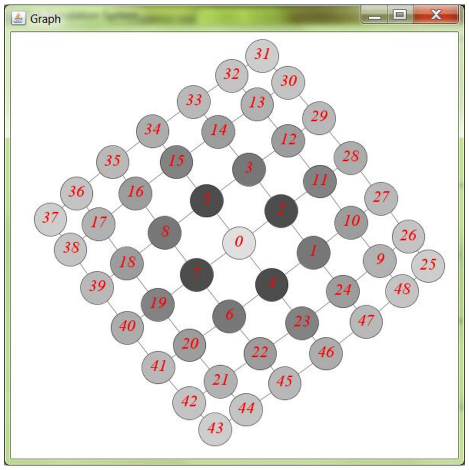


Figure 6.13: Entire absorbed energy distribution chart on a graph with energy injected from graph center.


(a) t=0


(b) t=1.


(c) t=5.

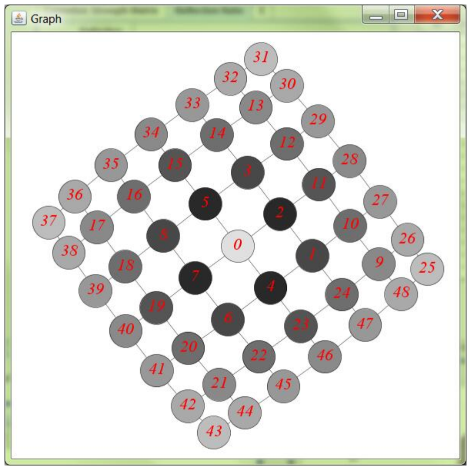

(d) t=20.

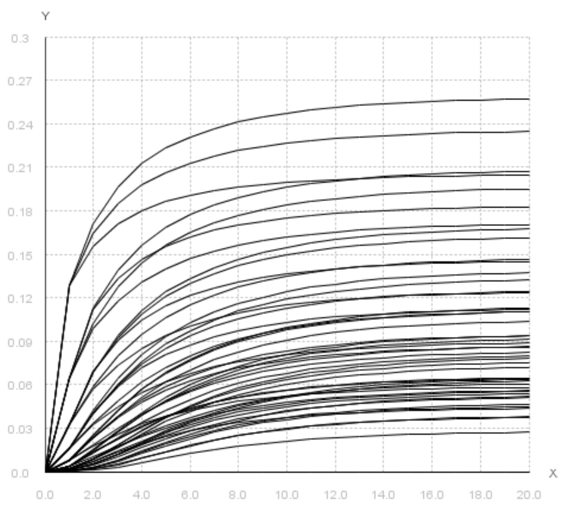
Figure 6.14: Entire absorbed energy distribution in graph with energy injected from graph center.

6.2.2 Energy Injected into Vertex near Graph Corner

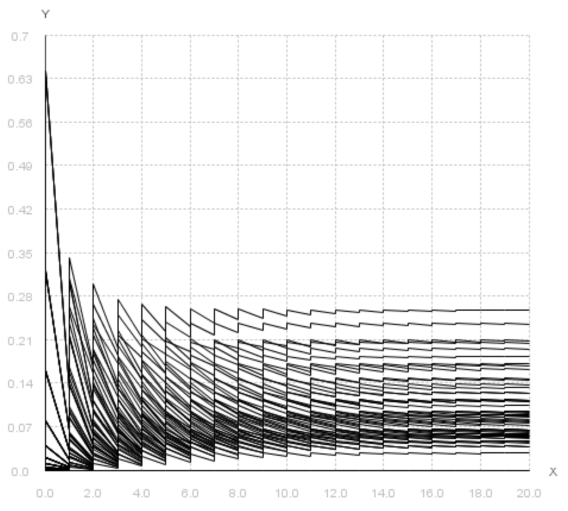
This subsection shows the pattern of energy distribution dynamics when the energy injected near graph corner. We set the runtime arguments as start vertex 36 and injected energy 5.

Figure 6.15 and Figure 6.16 show the results of this simulation. In Figure 6.15, the total contained energy dynamics for forty-eight vertices present lots of different oscillations. Figure 6.16 shows there are more irregularities in the whole energy flow process. This is due to the vertex thirty-six is non-symmetric vertex of the graph. Thus, the connection strength vector for this vertex distributed heterogeneously. Therefore, the received energy for each vertex at any time step will be distributed non-symmetrically on the graph.

Figure 6.18 shows the absorbed energy distribution at four different time steps. In this simulation, the average absorbed energy for each vertex which is 0.103 and the standard deviation of absorbed energy which is 0.056. Thus, the relative standard deviation of absorbed energy which is $(0.056/0.103) * 100\% = 54.4\%$. So we consider that the energy is distributed unevenly on the graph.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.15: Energy distribution dynamics on a graph with energy injected from a vertex near graph corner.

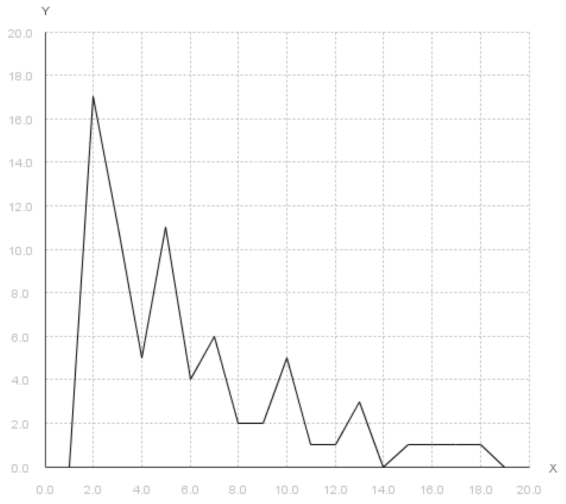


Figure 6.16: Irregularity in energy dynamics of the graph with energy injected near graph corner.

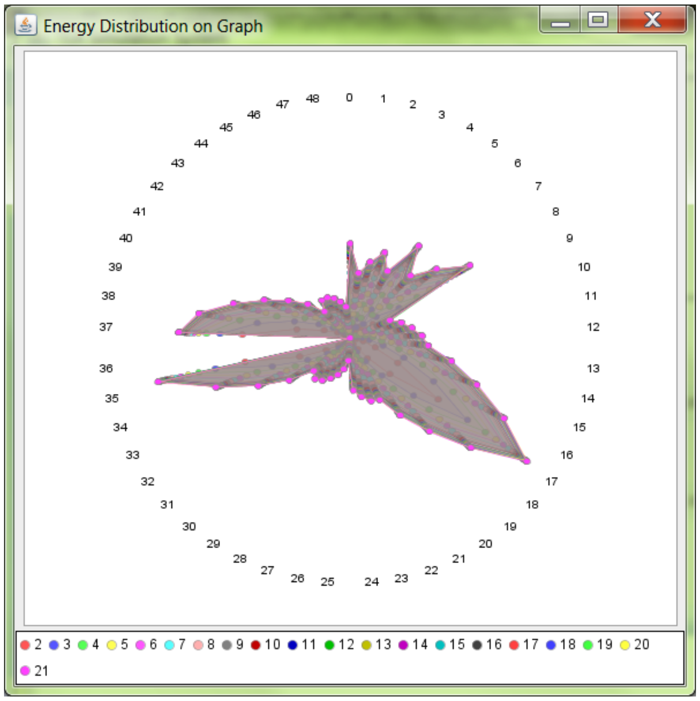
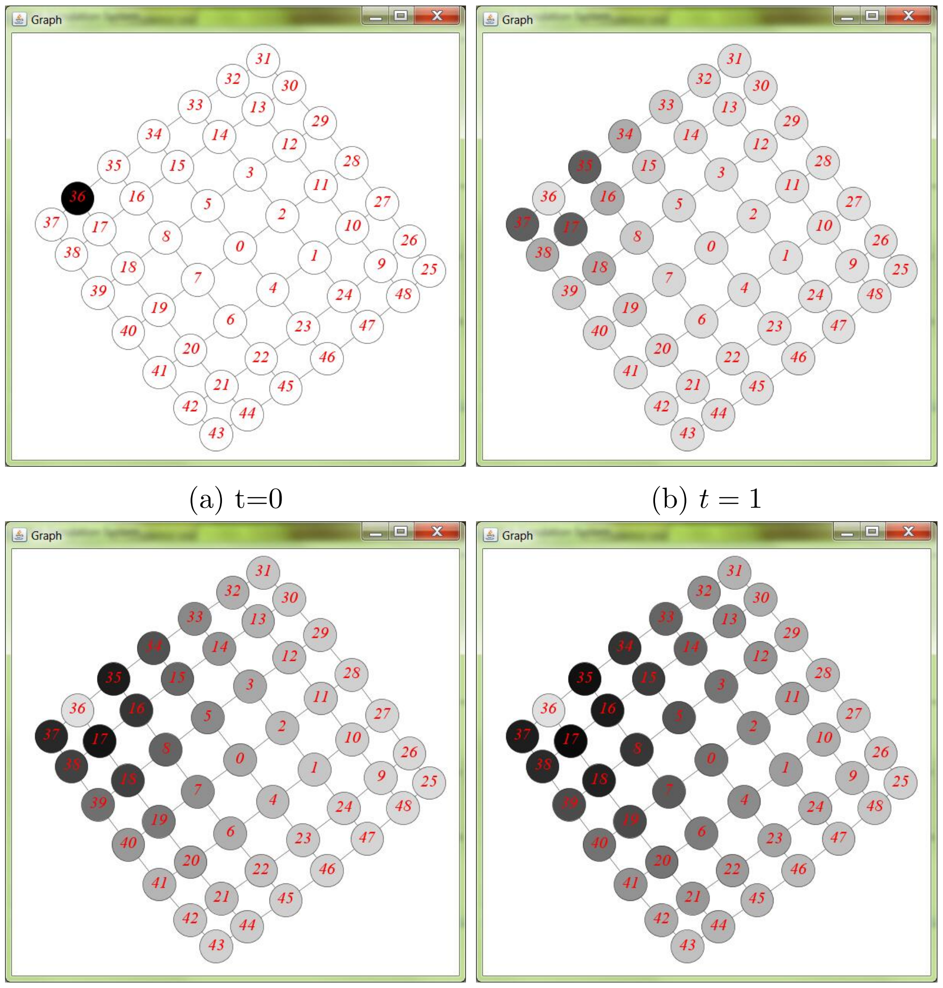


Figure 6.17: Entire absorbed energy distribution chart on a graph with energy injected near graph corner.


(a) t=0 (b) $t = 1$

(c) $t = 5$ (d) $t = 20$

Figure 6.18: Entire absorbed energy distribution on a graph with energy injected from a vertex near graph corner.

6.2.3 Remarks

In this section, we use paired comparison simulations to illustrate the impacts of energy distribution dynamics through injecting energy location. By comparing the results between the two simulations, one is based on forty-nine vertex homogeneous lattice graph with homogeneously distributed reflection ratios and energy injected from center graph, while the other is base on forty-nine vertex homogeneous lattice graph with homogeneously distributed reflection ratios and energy injected near graph corner, we can conclude:

- Location of energy injection has impacts on the variety of energy distribution dynamics;
- Location of energy injection has impacts on the irregularity in energy dynamic;
- Location of energy injection has impacts on the unevenly absorbed energy distribution;

6.3 Impacts of Heterogeneous Network Structure

In the previous section, we exhibit the dynamics of energy distribution in homogeneous lattice graph. However, with heterogeneous network structure, the energy distribution dynamics shows variety of different patterns. In this section, we will discuss the dynamics of energy distribution affected by heterogeneity of network structure.

We consider two ways to affect the heterogeneous of network structure in this section. One is by adjusting the length of connections in a homogeneously distributed graph, while the other is by adjusting the number of vertices in a homogeneously distributed graph. In order to demonstrate the behaviors of these two kinds heterogeneously distributed network, we design two simulations based on each kind of heterogeneous network structure. Then, we compare the result of each simulation with the result of homogeneous network structure in subsection 6.2.1 to make a conclusion. Specifically, we assume the reflection ratio for each vertex in these two simulations to be 0.8 and the runtime arguments as start vertex to be 0 and injected energy to be 5.

This section consists of two parts: subsection 6.3.1 exhibits impacts of heterogeneously distributed network structure by adjusting the length of the connections; subsection 6.3.2 exhibits impacts of heterogeneously distributed network structure by removing vertex.

6.3.1 Heterogeneous Length Distribution

In this section, we will show the behavior of heterogeneous length distributed network structure. This impacting factor means the distance of each connection on the graph is not even. Figure 6.19 shows the graph of this simulation. In this graph, we divide the vertices of forty-nine lattice graph into two groups: group I and group II, and define the length of each connection on the graph by following rules:

- For each connection that connects two vertices in the same group, such as connection between vertex 26 and vertex 27, the distance of this kind of connection is 1;
- For each connection that connects two vertices in the different groups, such as connection between vertex 27 and vertex 28, the distance of this kind of connection is 5.

Based on these two rules, we can calculate the connection strength matrix by the Algorithm 1 and build the TEA model.

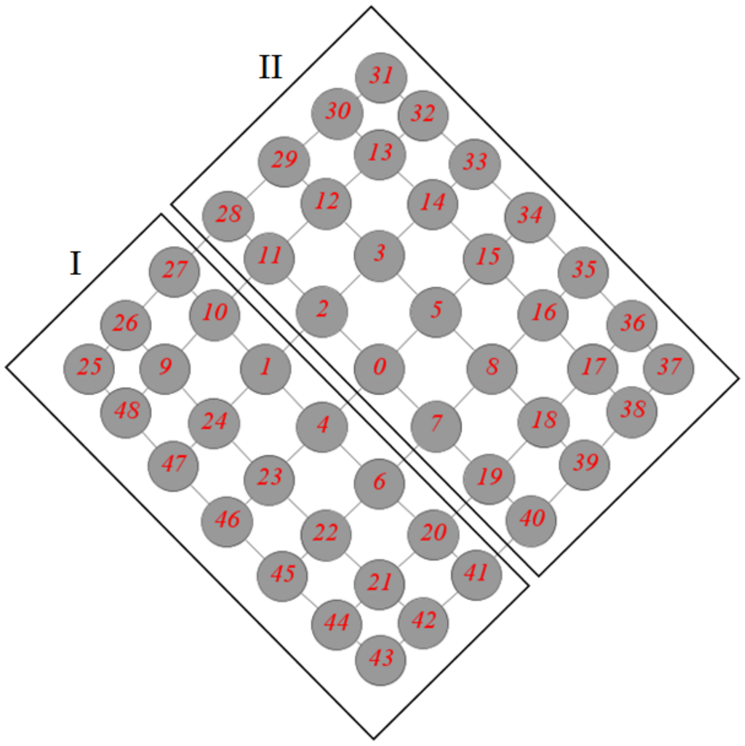
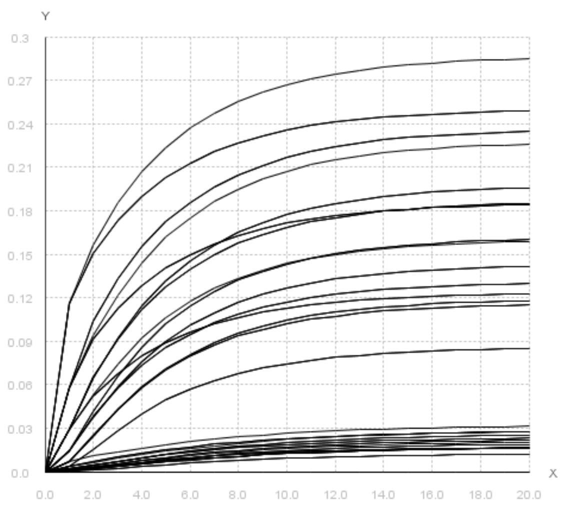
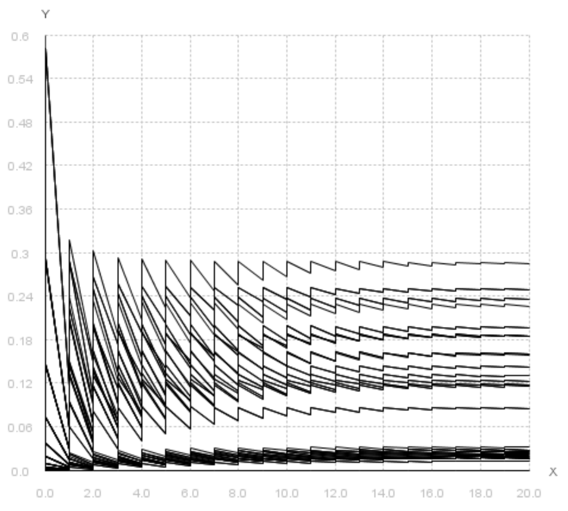


Figure 6.19: A lattice graph with forty-nine vertices heterogeneously length distributed.

Figure 6.20 and Figure 6.21 show the results of energy distribution dynamics in this simulation. In Figure 6.20, the energy distribution dynamics of forty-eight vertices present lots of different oscillations. Figure 6.21 shows there are a lot of irregularities in the whole process. These are because the factor of heterogeneous length distribution changes the structure of graph and makes the connection strength matrix non-symmetric. Therefore, the received energy by each vertex at any time step is distributed non-symmetrically on the graph.



(a) Entire absorbed energy dynamics.



(b) Total absorbed energy dynamics.

Figure 6.20: Energy distribution dynamics on a graph with heterogeneously length distributed.

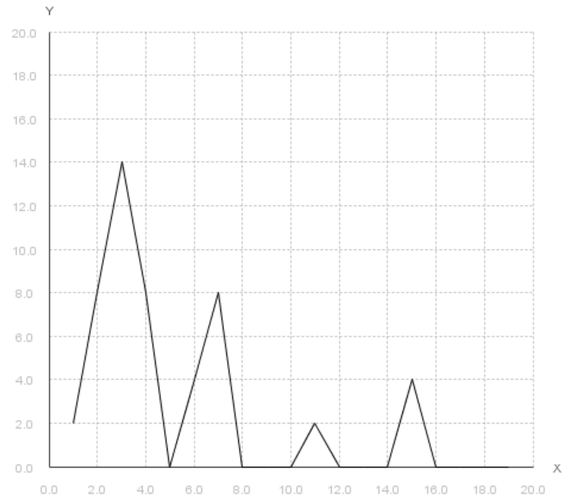


Figure 6.21: Irregularity in energy dynamics of a graph with heterogeneously length distributed.

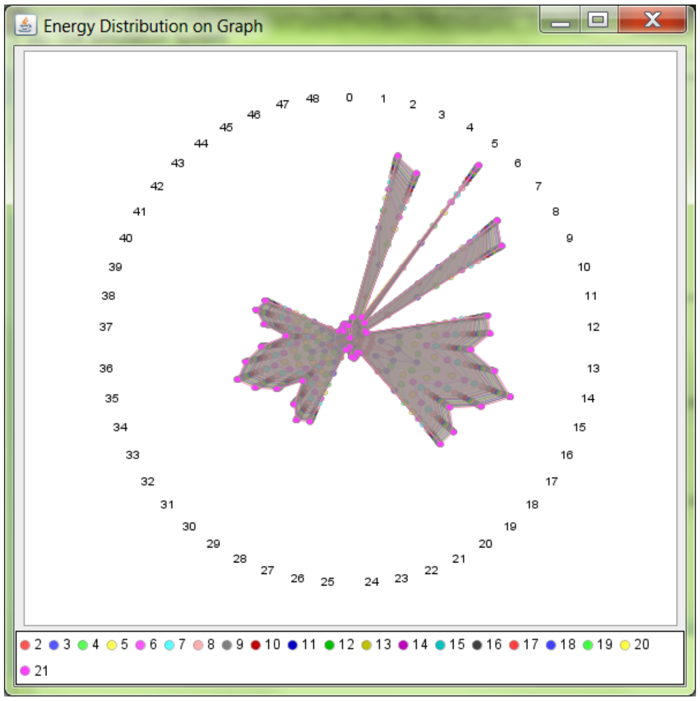
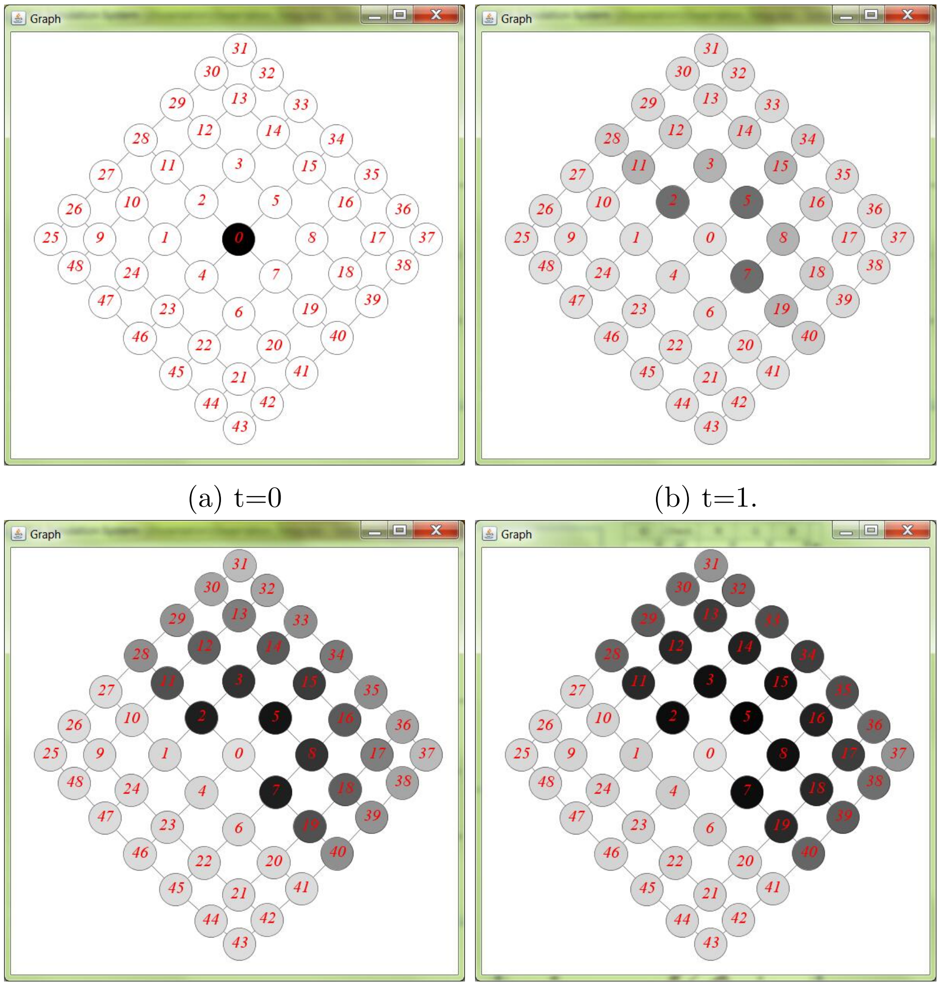


Figure 6.22: Entire absorbed energy distribution chart on a graph with heterogeneously length distributed.

Figure 6.23 shows the absorbed energy distribution at four different time steps.

In this simulation, the average absorbed energy for each vertex is 0.103 and the standard deviation of absorbed energy is 0.084. Thus, the relative standard deviation of absorbed energy which is $(0.084/0.103) * 100\% = 81.6\%$. So we consider that the energy is distributed unevenly on the graph.



(a) t=0 (b) t=1. (c) t=5 (d) t=20

Figure 6.23: Entire absorbed energy distribution on a graph with heterogeneously length distributed.

6.3.2 Heterogeneous Vertices Distribution

In this section, we will show the behaviors of heterogeneous vertices distributed structure. This impacting factor stands for the vertices distributed unevenly in the graph. As Figure 6.24 shows, we consider this network structure by removing vertex

8 out of the forty-nine homogeneous lattice graph while keeping other properties the same as the simulation in subsection 6.2.1.

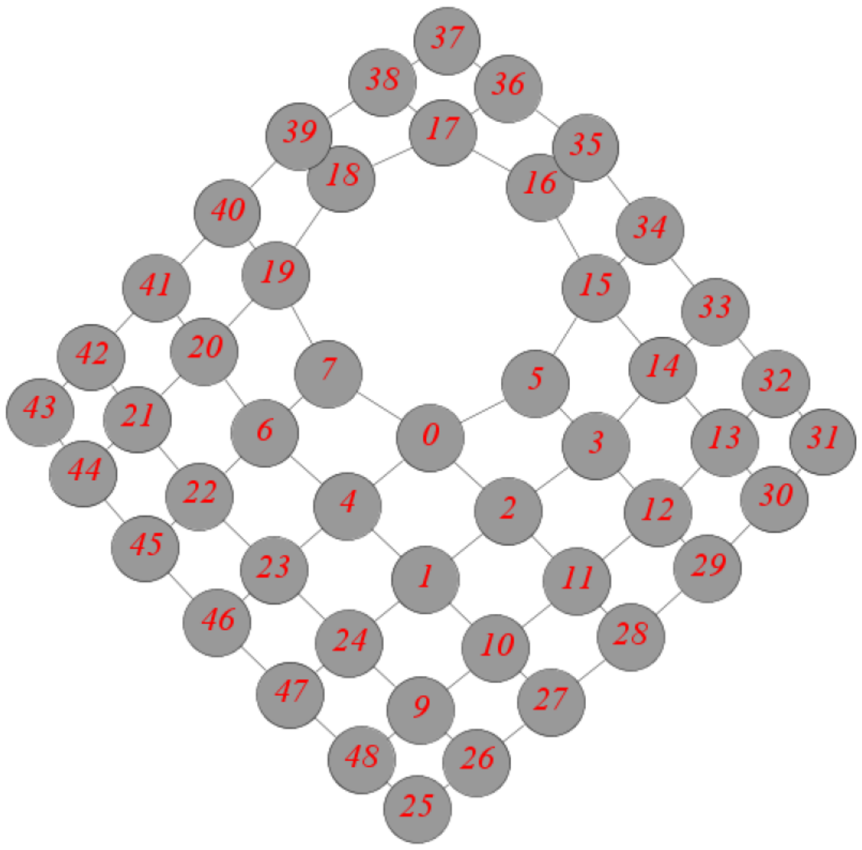
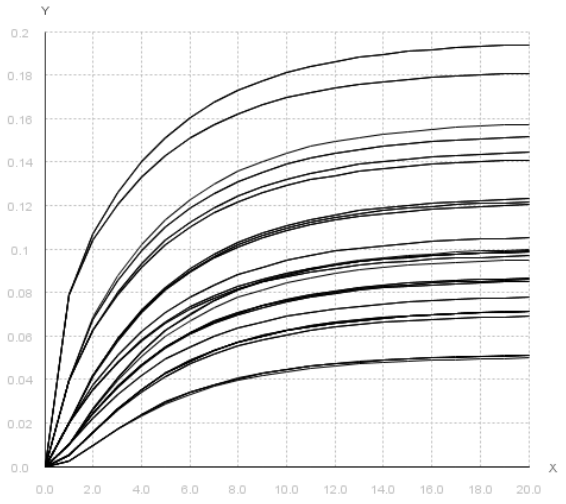
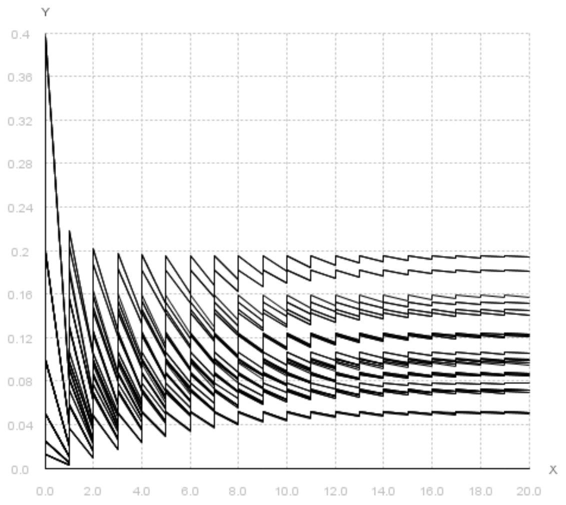


Figure 6.24: A lattice graph with forty-eight vertices heterogeneously distributed.

Figure 6.25 and Figure 6.26 show the results of energy distribution dynamics in this simulation. In Figure 6.25, the entire absorbed energy dynamics for forty-seven vertices present lots of different oscillations. Figure 6.26 shows there are more irregularities during the whole process. These are because this impacting factor changes the structure of graph. Thus, the connection strength matrix is non-symmetrical. Therefore, the distribution of received energy at any time step is distributed non-symmetrically on the graph.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.25: Energy distribution dynamics on a graph with heterogeneously vertices distributed.

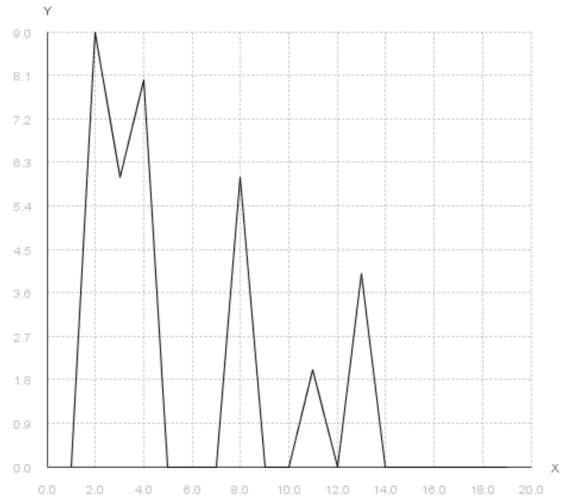


Figure 6.26: Irregularity in energy dynamics on a graph with heterogeneously vertices distributed.

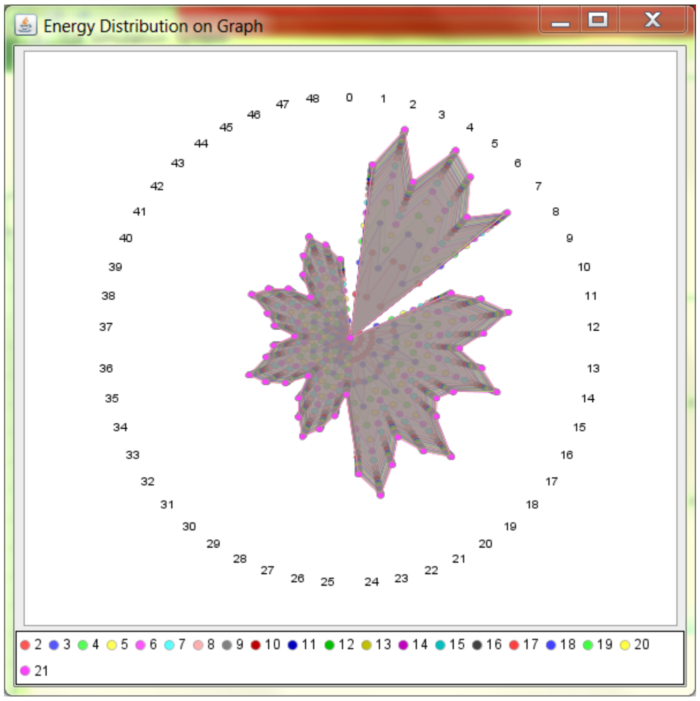
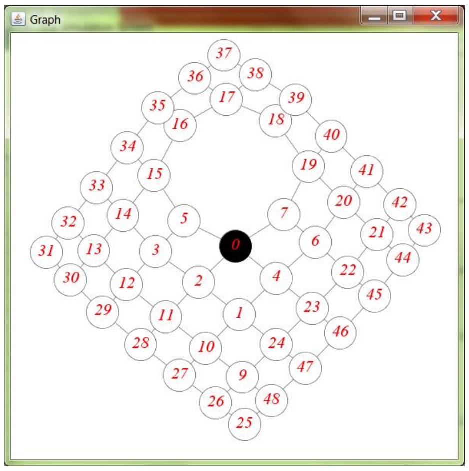
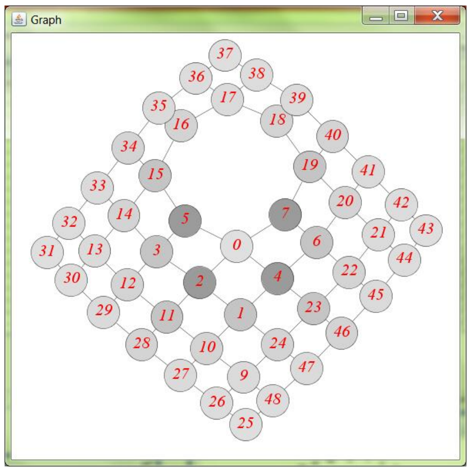


Figure 6.27: Entire absorbed energy distribution chart on a graph with heterogeneously vertices distributed.

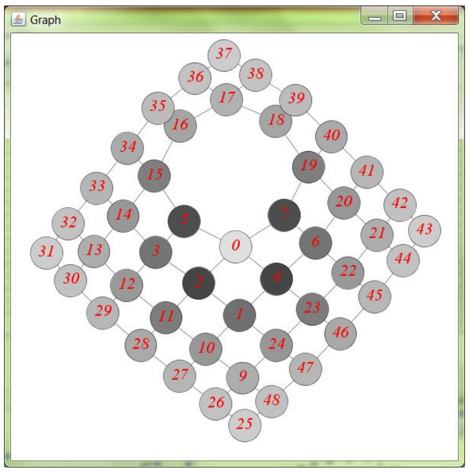
Figure 6.28 shows the entire absorbed energy distribution at four different time steps. In this simulation, the average absorbed energy for each vertex which is 0.105 and the standard deviation of absorbed energy which is 0.038. Thus, the relative standard deviation of absorbed energy which is $(0.038/0.105) * 100\% = 36.2\%$. So we consider that the energy is distributed evenly on the graph.



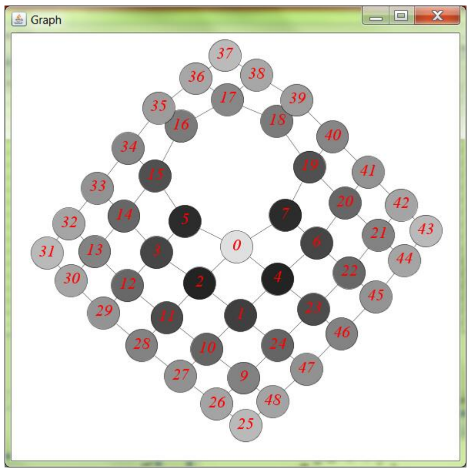
(a) t=0



(b) t=1.



(c) t=5



(d) t=20

Figure 6.28: Entire absorbed energy distribution on a graph with heterogeneously vertices distributed.

6.3.3 Remarks

In this section, we present two simulations of heterogeneously distributed network structure, one is by adjusting the length of connections while the other is by removing one vertex out of the graph. By comparing the results of these two simulations to the result of homogeneously distributed network structure in subsection 6.2.1, we can conclude:

- Both of the two factors, heterogeneous length distribution and heterogeneous vertices distribution, can change the network structure;
- Both of the two factors have impacts on increasing variety of energy distribution dynamics;
- Both of the two factors have impacts on increasing irregularity in the energy dynamics;
- Heterogeneously length distribution has impacts on the unevenly energy distribution.

6.4 Impacts of Heterogeneous Reflection Ratios Distribution

The last impacting factor of energy distribution dynamics on the graph we will discuss in this dissertation is heterogeneous reflection ratios distribution. In order to conclude the impacts of this reflection ratio distributed factor, we design a simulation based on forty-nine vertex homogeneous lattice graph with heterogeneously distributed reflection ratio and energy injected from center graph. We will compare the result of this simulation with forty-nine vertex homogeneous lattice graph with the simulation in sub section 6.2.1.

Heterogeneously distributed reflection ratios graph means, as what the name implies, a graph of reflection ratios distributed unevenly. In this simulation, we define the reflection ratio for each vertex except start vertex to be either 0.7 or 0.9; the average value of these reflection ratios is 0.8. The reflection ratio for each vertex on the graph is shown in Table 6.1

Vertex No.	0	1	2	3	4	5	6	7	8	9
Reflection Ratio	0.8	0.7	0.7	0.7	0.7	0.7	0.9	0.7	0.7	0.7

Vertex No.	10	11	12	13	14	15	16	17	18	19
Reflection Ratio	0.9	0.7	0.7	0.7	0.7	0.7	0.9	0.7	0.7	0.7

Vertex No.	20	21	22	23	24	25	26	27	28	29
Reflection Ratio	0.7	0.7	0.7	0.7	0.7	0.9	0.9	0.9	0.9	0.9

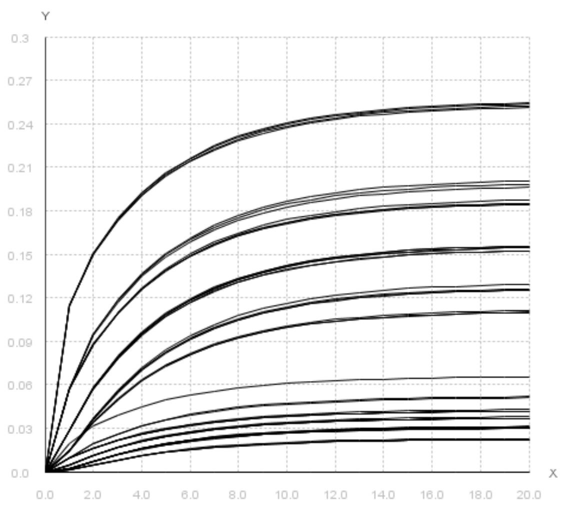
Vertex No.	30	31	32	33	34	35	36	37	38	39
Reflection Ratio	0.9	0.9	0.9	0.7	0.9	0.9	0.9	0.9	0.9	0.7

Vertex No.	40	41	42	43	44	45	46	47	48	49
Reflection Ratio	0.9	0.9	0.9	0.9	0.9	0.7	0.9	0.9	0.9	0.9

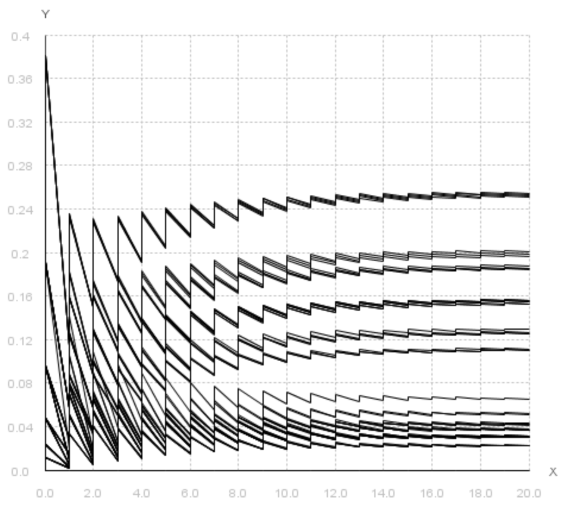
Table 6.1: Heterogeneous reflection ratios distribution.

6.4.1 Impacts on Time Evolution

Figure 6.29 and Figure 6.30 show the results of energy distribution dynamics in this simulation. In Figure 6.29, the energy distribution dynamics of forty-eight vertices present variety of different oscillations. Figure 6.30 shows there exists a lot of irregularities in the whole process. This is due to the heterogeneously distributed reflection ratios on graph. Since there are not all of the symmetric pair of vertices on the graph have same ability of reflecting energy, the reflected energy distribution on graph exhibits non-symmetric behaviors. Therefore, the energy distribution dynamics on graph exhibits so many different behaviors.



(a) Entire absorbed energy dynamics.



(b) Total contained energy dynamics.

Figure 6.29: Energy distribution dynamics on a graph with reflection ratio heterogeneously distributed.

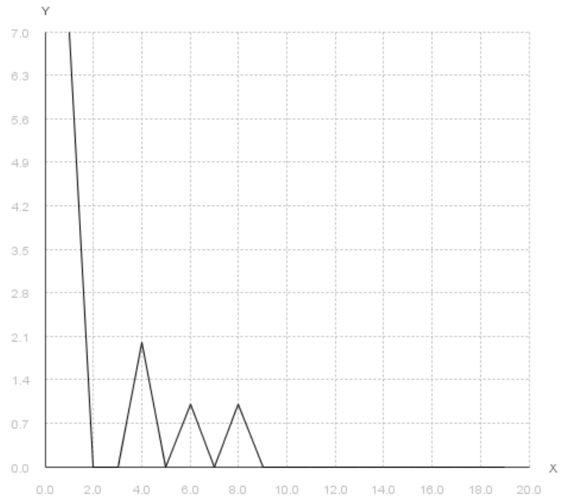


Figure 6.30: Irregularity in energy dynamics of a graph with reflection ratio heterogeneously distributed.

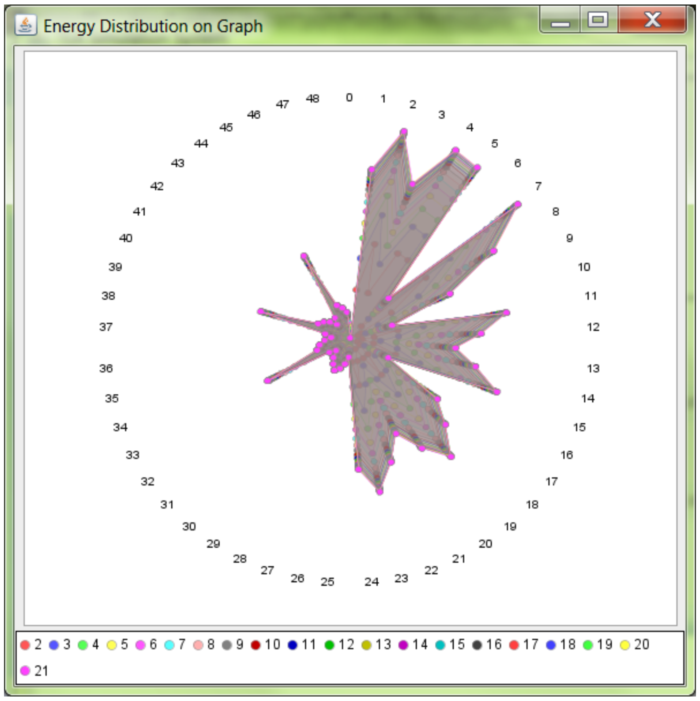
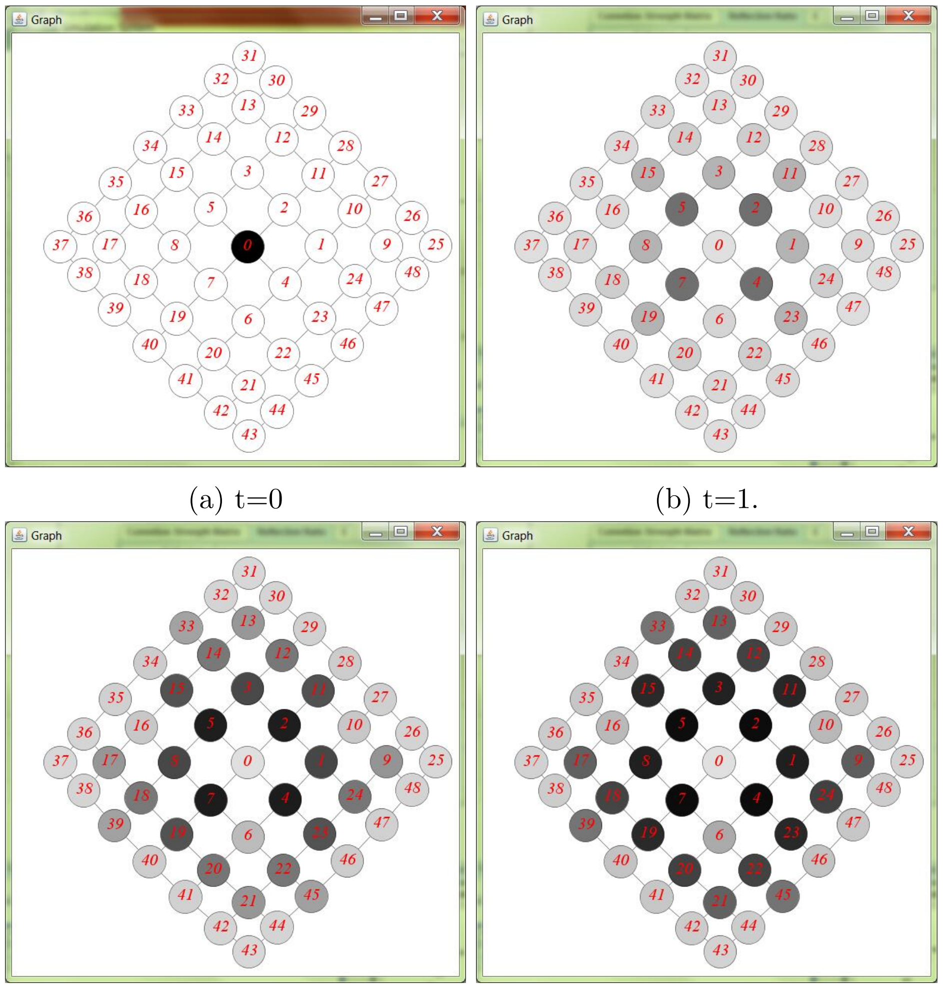


Figure 6.31: Entire absorbed energy distribution chart on a graph with reflection ratio heterogeneously distributed.

6.4.2 Impacts on Energy Distribution

Figure 6.32 shows the entire absorbed energy distribution at four different time steps. In this simulation, the average absorbed energy for each vertex is 0.103 and the standard deviation of absorbed energy is 0.076. Thus, the relative standard deviation of absorbed energy which is $(0.076/0.103) * 100\% = 73.8\%$. So we consider the energy is distributed unevenly on the graph.



(a) t=0 (b) t=1. (c) t=5 (d) t=20

Figure 6.32: Entire absorbed energy distribution on a graph with reflection ratio heterogeneously distributed.

6.4.3 Remarks

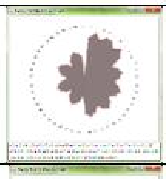
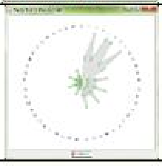
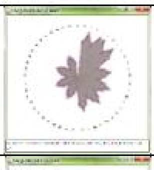
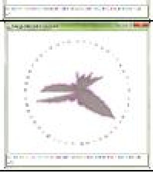
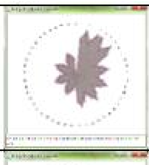
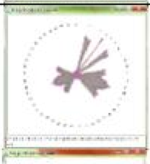
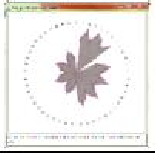
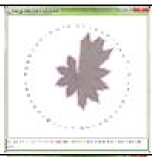
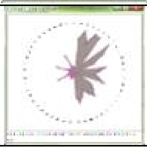
In this section, we present a simulation of heterogeneously distributed reflection ratios network structure. By comparing the result of this simulation to the result of homogeneously distributed reflection ratios graph in subsection 6.2.1, we can conclude:

- Heterogeneous reflection ratios distribution has impact on the variety of energy distribution dynamics;
- Heterogeneous reflection ratios distribution has impact on increasing the irregularities in energy dynamics;
- Heterogeneous reflection ratios distribution has impact on the unevenly energy distribution;

6.5 Summary

In this section, we introduce four kinds of impacting factors which affect the evolution of energy distribution on the graph. Table 6.33 summaries the impacts of those factors on the energy distribution dynamics of the graph. Although we summarize only four kinds of impacting factors in this dissertation, the combination of these factors could produce more complex result of energy distribution dynamics on the graph. In the next chapter, we will use TEA to simulate the energy distribution dynamics in a real power system.

Figure 6.33: Summaries of energy distribution dynamics affected by different impacting factors.

	Reflection Ratio	
Higher Reflection Ratio	1. More time will spend on the energy dynamics from start to converge; 2. More chaotic the oscillations of energy dynamics; 3. Energy prefers to distribute evenly.	
Lower Reflection Ratio	1. Less time will spend on the energy dynamics from start to converge; 2. Less chaotic the oscillations of energy dynamics; 3. Energy prefers to distribute unevenly.	
	Energy Injected Location	
Energy Injected into Graph Center	1. Less variety of energy dynamics trajectories; 2. Less number of crossovers; 3. Energy prefers to distribute evenly.	
Energy Injected into Vertex near Graph Corner	1. More variety of energy dynamics trajectories; 2. More irregularity of dynamics trajectories; 3. Energy prefers to distribute unevenly.	
	Network Structure	
Homogeneous Distribution Structure	1. Less variety of energy dynamics trajectories; 2. Less irregularity of dynamics trajectories; 3. Energy prefers to distribute evenly.	
Heterogeneous Distribution Structure	1. More variety of energy dynamics trajectories; 2. More irregularity of dynamics trajectories; 3. By factor of heterogenous length distribution, energy prefers to distribute unevenly.	 
	Reflection Ratios Distribution	
Homogeneous Reflection Ratios Distribution	1. Less variety of energy dynamics trajectories; 2. Less irregularity of dynamics trajectories; 3. Energy prefers to distribute evenly.	
Heterogeneous Reflection Ratios Distribution	1. More variety of energy dynamics trajectories; 2. More irregularity of dynamics trajectories; 3. Energy prefers to distribute unevenly.	

Chapter 7

Comparative Studies

In the previous chapter, we explore the energy distribution characteristics in a latter graph based on TEA. In this chapter, we will compare the energy distribution based on TEA with the one evaluated based on power system dynamics simulations in a power grid to validate the efficacy of TEA-based energy analysis.

7.1 Energy Flow Analysis Using Power System Model

Similar to the lattice graph as shown in Figure 6.1 and 6.10, a homogeneous power grid is constructed. As shown in Figure 7.1, the power grid has eighty-four lines and forty-nine buses, which is the same as previous simulation. Each bus is connected to a generator and a load.

In order to simulate the dynamics of the described power system in this chapter, we make the following assumptions:

- 1. Each generator is represented by a classical model, which is a constant voltage source behind transient reactance;
- 2. Governor action is not considered in this simulation while mechanical torque is assumed to be constant;

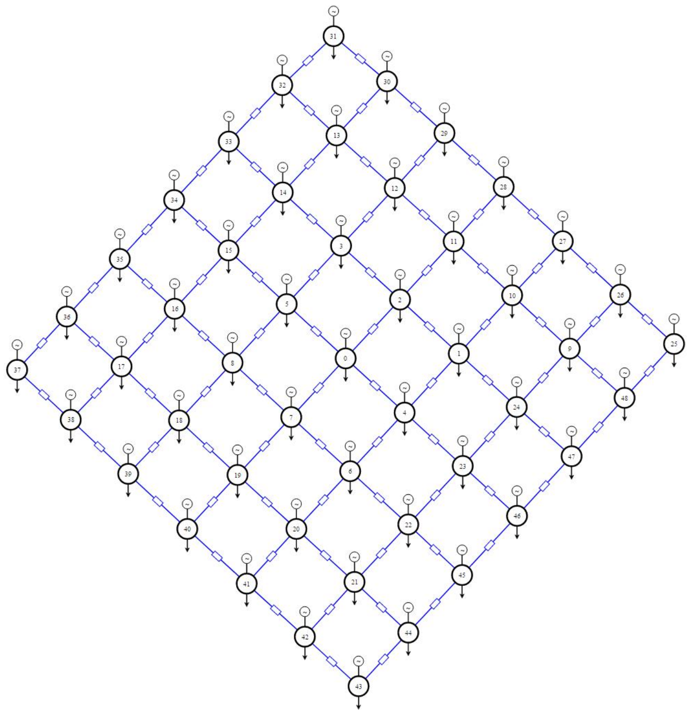


Figure 7.1: Forty-nine buses power system.

- 3. Damping coefficients are neglected;
- 4. Power consumption is neglected in transmission lines.

Under the above assumptions, the motion of the ith generator is described by the following differential equations in using centre of angle (COA) reference[40]:

$$M_i \dot{\omega}_i = P_{mi} - P_{ei} - \frac{M_i}{M_T} P_{COA} \tag{7.1}$$

$$\dot{\theta}_i = \omega_i \tag{7.2}$$

where

$$P_{ei} = \frac{E_i V_i sin(\theta_i - \phi_i)}{x'_{\mathrm{d}i}} \tag{7.3}$$

$$M_T = \sum_{t=1}^{m} M_i \quad , \quad P_{COA} = \sum_{t=1}^{m} (P_{mi} - P_{ei}) \tag{7.4}$$

In the previous equations, m is amount of generators in this power system, M_i is inertia constant of generator i, ω_i is the angular velocity of generator i in COA reference, θ_i and ϕ_i are angles of generator i and bus i with respect to COA, E_i is the internal voltage of generator i, V_i is the voltage at bus i, P_{mi} stands for the mechanical power input to generator i and P_{ei} stands for the electrical power generated by generator i. By the definition of COA variables [40], we have:

$$\sum_{i=1}^{m} M_i\theta_i = 0 \quad \text{and} \quad \sum_{i=1}^{m} M_i\omega_i = 0 \tag{7.5}$$

For each load model in this power system, we assume both active and reactive powers, P_{li} and Q_{li}, is given.

As we assumed there is no lossless in the power transmission, we have the following power flow equations. Let

$$\begin{aligned} g_{1i} &= \frac{E_i V_i sin(\phi_i - \theta_i)}{x'_{\mathrm{d}i}} \\ g_{2i} &= \sum_{j=1}^{n} V_i V_j B_{ij} sin\phi_{ij} \\ g_{3i} &= \frac{V_i^2 - E_i E_j cos(\theta_i - \phi_i)}{x'_{\mathrm{d}i}} \\ g_{4i} &= \sum_{j=1}^{n} V_i V_j B_{ij} cos\phi_{ij} \end{aligned} \tag{7.6}$$

where $B_{ij} = Im[Y_{bus}(i,j)]$, Y_{bus} is the admittance matrix of the network. The active power injected into the network from bus i is

$$P_i = g_{1i} + g_{2i} \tag{7.7}$$

The reactive power injection at bus i is

$$Q_i = g_{3i} + g_{4i} \tag{7.8}$$

Thus, the power flow equations at bus i can be written as

$$P_i + P_{li} = 0 \tag{7.9}$$

and

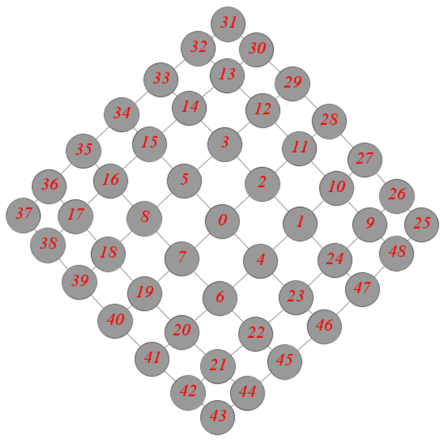
$$Q_i + Q_{li} = 0 \tag{7.10}$$

7.2 Energy Flow Analysis Using TEA model

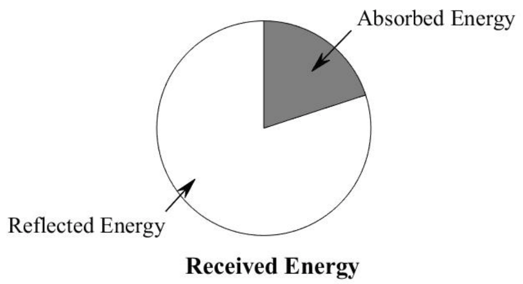
Before we introduce our comparison experiments, it is necessary to review our TEA model which will be used in the simulations. The TEA is a 8-tuples $\mathcal{A} = (\mathcal{N}, T, r, v_0, \pi, \mathcal{S}, \Sigma, \tau)$ where:

- $\mathcal{N} = \{\mathcal{V}, \mathcal{E}, B\}$ is a forty-nine vertex homogeneous lattices graph. In this graph, each connection is one.
- $T \in \mathbb{R}^{49\times 49}$ is the connection strength matrix of the network;
- $r \in \mathbb{R}^{49\times 1}$ is the reflection ratio vector of the graph where each element is 0.8;
- $v_0 \in V(\mathcal{G})$ is the start vertex where energy is injected in;
- $\pi \in \mathbb{R}$ is the amount of energy which will be injected into the vertex v_0;
- $\mathcal{S}$ contains two states where -1 means fast action while 1 means slow action;
- Σ records the energy status of each vertex in TEA;
- τ is a function that maps the current energy status into next energy status.

The test case in this simulation is shown in Figure 7.2.



(a) Forty-nine vertex homogeneous lattice graph with the length of each connection is one.



(b) Model of each vertex where we assume the reflection ratio is 0.8.

Figure 7.2: Model of comparative study.

7.3 Comparison between TEA and Power System Dynamic Simulation

In this experiments, we compare the energy distribution evaluated based on power system dynamic simulation with those based on TEA when the energy is injected from different locations. In power system dynamic simulations, the energy injection is triggered by applying a fault at a specific bus in the grid at $t = 0s$ and later cleared at $t_c = 0.1s$.

7.3.1 Distributions of Energy Injected from Center Bus

We first compare the energy distribution evaluated based on power system dynamic simulations with those based on TEA when the energy is injected into the grid from center bus 0 as shown in Figure 7.3.

We can see the results of these two simulations in Figure 7.4. It can be concluded that TEA and power system dynamic simulations provide the consistent evaluation results when the energy is injected from the center bus. As shown in Figure 7.4, the energy profile provided by power system dynamic simulations is almost the same as the one provided by TEA. Specifically, based on the TEA evaluation results as shown in Figure 7.4b, the buses in the power grid can be generally classified into three groups (groups $a - c$) in terms of their received energy: group a is composed of vertices 1-8; group b consists of vertices 9-26; and group c is made up of vertices 27-48. Based on the results of power system dynamic simulations as shown in Figure 7.4a, the buses in the grid can be also be classified into three groups (groups $a' - c'$) in terms of their received energy: group a' is composed of vertices 1-8; group b' consists of vertices 9-26; and group c' is made up of vertices 27-48. By comparing these groups one by one, it is found that one of these three groups a, b, c not only has the same energy profile as the one corresponding to the those three groups a', b', c', but also has the same buses. For example, both group a and group a' are composed of four peaks, which are vertices 2, 4, 5 and 7, and the energy received from these four vertices forms the same profile. Thus, when the energy is injected into the grid at the center bus 0, the two methods provide consistent results of energy distribution.

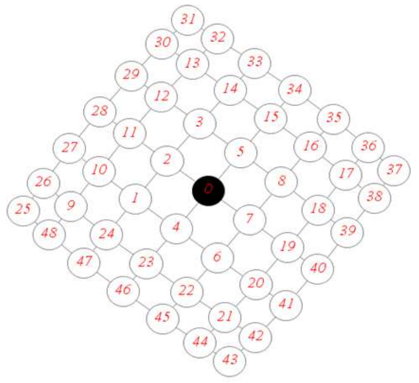
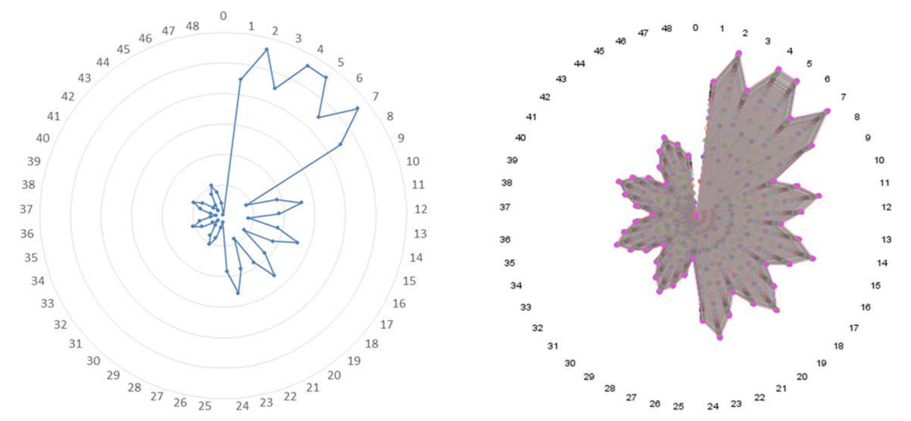


Figure 7.3: Power fault happened at the center of the power network.



(a) Energy distribution result by power system dynamic simulation. (b) Energy distribution result by TEA simulation.

Figure 7.4: Energy distribution results of energy injected from center bus.

7.3.2 Distributions of Energy Injected from Corner Bus

We further compare the energy distribution evaluated based on power system dynamic simulations with those of TEA when the energy is injected into the grid from corner bus 37 as shown in Figure 7.3.

We can see the results of these two simulations in Figure 7.4. It can be concluded that TEA and power system dynamic simulations provide the highly similar evaluation results when the energy is injected from the corner bus. As shown in Figure 7.6, the energy profile provided by power system dynamic simulations is very similar to the one provided by TEA. Specifically, based on the TEA evaluation results as shown in Figure 7.6b, all buses in the power grid can be classified into four groups (groups $a - d$) in terms of their received energy: group a is composed of vertices 0-11; group b consists of vertices 12-25; group c is made up of vertices 26-36; and group d is composed of vertices 38-48. Based on the results of power system dynamic simulations as shown in Figure 7.6a, all buses in the power grid can also be classified into four groups (groups $a' - d'$) in terms of their received energy: group a' is composed of vertices 0-11; group b' consists of vertices 12-25; group c' is made up of vertices 26-36;

and group $d'$is composed of vertices 38-48. Comparing these groups one by one, we find that one of the four groups a, b, c, d not only has the similar energy profile to the corresponding one in those three groups a', b', c', d', but also has the similar buses. Thus, when the energy is injected into the grid at the corner bus 37, the two methods generated highly similar results of energy distribution.

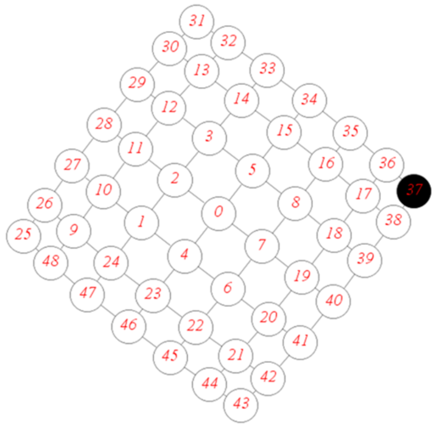
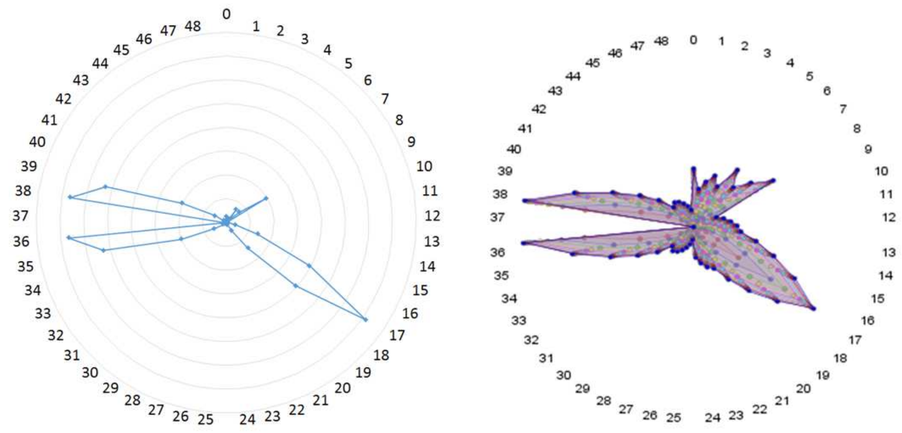


Figure 7.5: Power fault happened at a corner of the power network.



(a) Energy distributed result by power system dynamic simulation. (b) Energy distributed result by TEA simulation.

Figure 7.6: Energy distribution results of energy injected from corner bus.

7.4 Summary

In the preliminary study, we use the shortest distance between each pair of vertices to reflect the link relationship of them. This distance evaluation method may not correctly reflect the link relationship of two buses in the power system field. However, it still qualitatively describes the dependence of each pair of buses.

In this chapter, we study the energy injected into graph center and graph corner by two different approaches. By comparing the distribution evaluated based on TEA with the one evaluated based on power system dynamic simulations, we find the results obtained by two different approaches are qualitatively consistent with each other. More specifically, the result of energy injected into graph center by TEA-based energy analysis is more accurate than the result of energy injected into a graph corner by the same method. The reason for this is: when energy injected into the graph center, the shortest distance between each pair of vertices consistent with the link relationship of them; when energy injected into a graph corner, the shortest distance method is still good, but with more discrepancy.

The results of these two experiments show that simple model of TEA may generate the similar results of complicate model in power system dynamic simulation. Moreover, TEA offers more insights of energy flow on graph. This approach could be useful for large scale power system analysis.

Chapter 8

Conclusions and Future Work

This chapter summarizes the contributions made in the related work and states some research directions for future work.

8.1 Conclusions

In this dissertation, we design the Transient Energy Automaton (TEA) to simulate the energy flow in a lattice graph. This TEA is a computational system that automates dynamics of energy exchanges on graph once triggered by disturbances similar to those used in the analysis of the dynamics of multi-machine power model system. The design and implementation of TEA is mainly based on cellular automata and graph visualization approach.

In order to simulate the dynamics of energy on the graph, TEA enables two fundamental actions and correlated rules of these two actions for each vertex in dynamics of energy flow in the graph. The two fundamental actions include: the slow action of a vertex to absorb the energy received from other vertices; and the fast action of a vertex to reflect the other part of energy to the other vertices. For simplifying and implementing these actions into the TEA model, we assume the couplings of these actions are governed by the following rules:

- For each vertex, the amount of energy received at a local site in the reflection action, which is defined as dispersion ratio, is a function of the graph geometry including the structure of lattice and value configuration of edges;

- The fraction of energy that can be absorbed by each vertex in the absorption action, which is defined as absorption ratio, is based on the capability of the vertex;

- We suppose at a particular time instant, all the vertices are in one same action, either fast action or slow action;

- Such rules are functions of the geometry and value configuration of graph and the property of vertices, which ultimately enable the TEA to create time evolution of energy distribution, that may affect the dynamics of the system.

The energy flow among vertices on graph and its time evolution are computed recursively using computational formula of the two actions under the governing rules.

We also design and realize our TEA simulation system for providing time-based energy distribution visualization services to the user. This system consists of three modules: controller module generates the TEA case data file; evolution simulator read the generated case data, accept runtime arguments and process the received data to output the energy distribution data; visualization module displays the evolution of energy distribution. The whole system is mainly developed by Java.

The simulation results show that with the simple rules, such as homogeneously or heterogeneously distributed network structures, TEA can produce varieties of complex energy distribution dynamics. Table 6.33 summaries the impacts of four factors on the energy distribution dynamics of the graph. Although we summarize only four kinds of impacting factors in this dissertation, the combination of these factors could produce more complex results of energy distribution dynamics on the graph.

At last, we compare the energy distribution evaluated based on TEA with the one evaluated based on power system dynamic simulations in a homogeneous power grid. The results of the two experiments, the one is energy injected into graph center while the other is energy injected into graph corner, show that the energy distribution obtained by two different approaches are qualitatively consistent with each other. This shows that simple model of TEA may generate the similar results of complicate model in power system dynamic simulation. Moreover, TEA offers more insights of energy flow on graph. This approach could be useful for large scale power system analysis.

8.2 Future Work

Unsynchronized action for each vertex at a time should be considered in a more realistic TEA model. Our model assumed all of the vertices are in one action, either in fast action or in slow action, at the same time. Moreover, during a fixed time interval, TEA supposes each vertex in the network will process one fast action and one slow action. In a realistic model, the actions of all vertices may not be synchronized. Thus, more realistic TEA model must incorporated with unsynchronized actions for all the vertices in the graph.

Variable model parameters should be implemented into a more realistic TEA model. In this dissertation, we simulate the energy flow on ideal models. This means we assume the values of the parameters used here, which include the reflection ratio for each vertex and the connection strength matrix of the network, remain the same in the whole process. More realistic models of vertices and network should consider the fluctuation of reflection ratio for each vertex and variable connection strength matrix for the TEA network over times.

More experiments on real power grid should be simulated by TEA-base energy analysis method. In our study, we have presented the comparison of the energy distribution evaluated based on TEA with the one evaluated based on power system dynamic simulations in a homogeneous power grid. However, this is an ideal power grid which is not the real power system. Furthermore, the simulation of TEA-based energy analysis is not performed on the real power system. Thus, more experiments based on real power grid, such as Taxes power grid, are needed to be simulated in order to validate the efficacy of TEA-based energy analysis.

Bibliography

[1] Jyotirmay Mathur, Narendra Kumar Bansal, and Hermann-Joseph Wagner. Dynamic energy analysis to assess maximum growth rates in developing power generation capacity: case study of india. *Energy Policy*, 32(2):281 – 287, 2004.

[2] Stein Keijzers. *Energy Consumption Analysis of Practical Programming Languages*. PhD thesis, Radboud University Nijmegen, 2014.

[3] R.S. Hartman. Frontiers in energy demand modeling. *Annual Review of Energy*, 4:433–466, 1979.

[4] J. A. Butts and G. S. Sohi. A static power model for architects. In *Proceedings 33rd Annual IEEE/ACM International Symposium on Microarchitecture. MICRO-33 2000*, pages 191–201, 2000.

[5] B. Friedman, T. A. Carter, M. V. Umansky, D. Schaffner, and B. Dudson. Energy dynamics in a simulation of LAPD turbulence. *Physics of Plasmas*, 19(10):102307, October 2012.

[6] Charles Xie. Interactive Heat Transfer Simulations for Everyone. *The Physics Teacher*, 50:237–240, 2012.

[7] X.-S. Yang and Y. Z. L. Yang. Cellular Automata Networks. *ArXiv e-prints*, March 2010.

[8] S. K. Lemieux A. A. Patel, E. T. Gawlinski and R. A. Gatenby. A cellular automaton model of early tumor growth and invasion: The effects of native tissue vascularity and increased anaerobic tumor metabolism. *J. theor. Biol.*, 213:315–331, 2001.

[9] J. Kari. Theory of cellular automata:a survey. *Theoretical Computer Science*, 334:3–33, 2005.

[10] Stephen Wolfram. Computation theory of cellular automata. *Communications in Mathematical Physics*, 96(1):15–57, 1984.

[11] Ansgar Kirchner, Katsuhiro Nishinari, and Andreas Schadschneider. Friction effects and clogging in a cellular automaton model for pedestrian dynamics. *Phys. Rev. E*, 67:056122, May 2003.

[12] S. Heike G. Martin and T. John J. A cellular automaton model of excitable media including curvature and dispersion. *Phys. Rev. E*, 247:1563–1566, March 1990.

[13] H.M. Byrne T. Alarcon and P.K. Maini. A cellular automaton model for tumour growth in inhomogeneous environment. *Journal of Theoretical Biology*, 225:257–274, June 2003.

[14] M. Fukui and I. Ishibashi. Traffic flow in 1d cellular automaton model including cars moving with high speed. *Journal of Physical Society of Japan*, 65(6):1868–1870, June 1996.

[15] A. Kirchner and A. Schadschneider. Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. *Physica A*, 312:260 276, February 2002.

[16] K. C. Clarke and L. J. Gaydos. Loose-coupling a cellular automaton model and gis: Long-term urban growth prediction for san francisco and washington/baltimore. *International Journal of Geographical Information Science*, 12(7):699–714, 1998.

[17] Sahotra Sarkar, Kelley A Crews-Meyer, Kenneth R Young, Christopher D Kelley, and Alexander Moffett. A dynamic graph automata approach to modeling landscape change in the andes and the amazon. *Environment and Planning B: Planning and Design*, 36(2):300–318, 2009.

[18] C. F. Lee M. D. Fricker D.Smith, J. Onnela and N. F. Johnson. Network automata: Coupling structure and function in dynamic networks. *Advances in Complex Systems*, 14(03):317–339, 2011.

[19] C. Marr and M.-T. Hütt. Topology regulates pattern formation capacity of binary cellular automata on graphs. *Physica A Statistical Mechanics and its Applications*, 354:641–662, August 2005.

[20] Carsten Marr and Marc-Thorsten Htt. Outer-totalistic cellular automata on graphs. *Physics Letters A*, 373(5):546 – 549, 2009.

[21] Giacobini M. Tomassini, M. and C. Darabos. Evolution and dynamics of small-world cellular automata. *Complex Systems*, (15):261 – 284, 2005.

[22] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Introduction to automata theory, languages, and computation, 2nd edition. *SIGACT News*, 32(1):60–65, March 2001.

[23] I. Herman, G. Melancon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, Jan 2000.

[24] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1998.

[25] X. Li, M. Javadi, B. Zhao, D. Wu, and J. N. Jiang. Evidence of significance of graph energy interlinks in topological form of the power grid. In *2015 5th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, pages 2596–2602, Nov 2015.

[26] Y. F. Hu. Efficient and high quality force-directed graph drawing. *The Mathematica Journal*, 663(10):37–71, 2005.

[27] Ulrik Brandes. *Layout of Graph Visualizations*. PhD thesis, Uni Konstanz, 1999.

[28] T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. *Software Practice Experience*, 21(11):1129–1164, 1991.

[29] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989.

[30] P. Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.

[31] G. E. Uhlenbeck and L. S. Ornstein. On the theory of the brownian motion. *Phys. Rev.*, 36:823–841, Sep 1930.

[32] W. T. Coffey and Yu P. Kalmykov. *The Langevin Equation: With Applications to Stochastic Problems in Physics, Chemistry and Electrical Engineering*. World Scientific, 2nd edition edition, 2003.

[33] Edward N. Lorenz. *Deterministic Nonperiodic Flow*, pages 25–36. Springer New York, New York, NY, 2004.

[34] GUANRONG CHEN and TETSUSHI UETA. Yet another chaotic attractor. *International Journal of Bifurcation and Chaos*, 09(07):1465–1466, 1999.

[35] Jinhu L, Tianshou Zhou, and Suochun Zhang. Chaos synchronization between linearly coupled chaotic systems. *Chaos, Solitons & Fractals*, 14(4):529 – 541, 2002.

[36] A. Khan and P. Singh. Chaos synchronization in lorenz system. *Applied Mathematics*, 6:1864–1872, 2015.

[37] Damei Li, Jun an Lu, and Xiaoqun Wu. Linearly coupled synchronization of the unified chaotic systems and the lorenz systems. *Chaos, Solitons & Fractals*, 23(1):79 – 85, 2005.

[38] Andrzej Stefanski, Tomasz Kapitaniak, and John Brindley. Dynamics of coupled lorenz systems and its geophysical implications. *Physica D: Nonlinear Phenomena*, 98(2):594 – 598, 1996. Nonlinear Phenomena in Ocean Dynamics.

[39] J. Bouttier, P. Di Francesco, and E. Guitter. Geodesic distance in planar graphs. *Nuclear Physics B*, 663(3):535 – 567, 2003.

[40] T. Athay, R. Podmore, and S. Virmani. A practical method for the direct analysis of transient stability. *IEEE Transactions on Power Apparatus and Systems*, PAS-98(2):573–584, March 1979.

[41] W. Vickrey. Counterspeculation, auctions and sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[42] H. S. Stern A. Gelman, J. B. Carlin and D. B. Rubin. *Bayesian Data Analysis*. Chapman Hall, 1995.

[43] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLOS ONE*, 9(6):1–12, 06 2014.

[44] C. Upson, T. A. Faulhaber, D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam. The application visualization system: a computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9(4):30–42, July 1989.

[45] Subhes C. Bhattacharyya and Govinda R. Timilsina. A review of energy system models. *International Journal of Energy Sector Management*, 4(4):494–518, 11 2010.

[46] G. Andersson, P. Donalek, R. Farmer, N. Hatziargyriou, I. Kamwa, P. Kundur, N. Martins, J. Paserba, P. Pourbeik, J. Sanchez-Gasca, R. Schulz, A. Stankovic, C. Taylor, and V. Vittal. Causes of the 2003 major grid blackouts in north america and europe, and recommended means to improve system dynamic performance. *IEEE Transactions on Power Systems*, 20(4):1922–1928, Nov 2005.

[47] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. De Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1/2):63+, 2008.

[48] Kindermann M. Beenakker, C.W.J. and Yu.V. Nazarov. Temperature-dependent third cumulant of tunneling noise. *Physical Review Letters, 90 (17)*, 90:17, 2003.

[49] K C Hoffman and D O Wood. Energy system modeling and forecasting. *Annual Review of Energy*, 1(1):423–453, 1976.

[50] Zoltn sik, Uli Fahrenberg, and Axel Legay. -continuous kleene -algebras for energy problems. *Electr. Proc. Theor. Comput. Sci*, 191:48–59, 9 2015.

[51] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jiří Srba. Energy games in multiweighted automata. In *Proceedings of the 8th International Conference on Theoretical Aspects of Computing*, ICTAC'11, pages 95–115, Berlin, Heidelberg, 2011. Springer-Verlag.

[52] Yuying Gao. *Cellular Automata in Financial Applications.* PhD thesis, USA, 2008. AAI3305677.

[53] S Bandini, G Mauri, and R Serra. Cellular automata: From a theoretical parallel computational model to its application to complex systems. *Parallel Computing*, 27(5):539 – 553, 2001. Cellular automata: From modeling to applications.

[54] Manfred Requardt and Saeed Rastgoo. The Structurally Dynamic Cellular Network and Quantum Graphity Approaches to Quantum Gravity and Quantum Geometry - A Review and Comparison. *J. Cell. Automata*, 10(5-6):341–392, 2015.

[55] S. Hoya White, A. Martn del Rey, and G. Rodrguez Snchez. Modeling epidemics using cellular automata. *Applied Mathematics and Computation*, 186(1):193 – 202, 2007.

[56] AALPEN A. PATEL, EDWARD T. GAWLINSKI, SUSAN K. LEMIEUX, and ROBERT A. GATENBY. A cellular automaton model of early tumor growth and invasion: The effects of native tissue vascularity and increased anaerobic tumor metabolism. *Journal of Theoretical Biology*, 213(3):315 – 331, 2001.

[57] Jarkko Kari. Theory of cellular automata: A survey. *Theoretical Computer Science*, 334(1):3 – 33, 2005.

[58] Martin Gerhardt, Heike Schuster, and John J. Tyson. A cellular automaton model of excitable media. *Physica D: Nonlinear Phenomena*, 46(3):392 – 415, 1990.

[59] T. Alarcn, H.M. Byrne, and P.K. Maini. A cellular automaton model for tumour growth in inhomogeneous environment. *Journal of Theoretical Biology*, 225(2):257 – 274, 2003.

[60] Ansgar Kirchner and Andreas Schadschneider. Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. *Physica A: Statistical Mechanics and its Applications*, 312(1):260 – 276, 2002.

[61] KEITH C. CLARKE and LEONARD J. GAYDOS. Loose-coupling a cellular automaton model and gis: long-term urban growth prediction for san francisco and washington/baltimore. *International Journal of Geographical Information Science*, 12(7):699–714, 1998. PMID: 12294536.

[62] David O'Sullivan. Graph-cellular automata: A generalised discrete urban and regional model. *Environment and Planning B: Planning and Design*, 28(5):687–705, 2001.

[63] H.-M. Groscurth, Th. Bruckner, and R. Kmmel. Modeling of energy-services supply systems. *Energy*, 20(9):941 – 958, 1995.

[64] Sergey V. Buldyrev, Roni Parshani, Gerald Paul, H. Eugene Stanley, and Shlomo Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, 04 2010.

[65] Paul M Anderson and Aziz A Fouad. *Power system control and stability*. John Wiley & Sons, 2008.

[66] John J Stevenson Grainger, William D John J Grainger, and William D Stevenson. *Power system analysis*. 1994.

[67] J. Gou, J. Liu, G. A. Taylor, C. S. Saunders, and Y. Liu. Complexity analysis of power system energy flow. In *2014 International Conference on Power System Technology*, pages 199–203, Oct 2014.

[68] M. Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 3nd edition edition, 2013.

[69] P. Mazur. On the theory of brownian motion. *Physica*, 25(1):149 – 162, 1959.

[70] Vladimir Nikiforov. The energy of graphs and matrices. *Journal of Mathematical Analysis and Applications*, 326(2):1472 – 1475, 2007.

[71] B. Guinot and M. Granveaud. Atomic time scales. *IEEE Transactions on Instrumentation and Measurement*, 21(4):396–400, Nov 1972.

[72] J. R. Mart, H. Ahmadi, and L. Bashualdo. Linear power-flow formulation based on a voltage-dependent load model. *IEEE Transactions on Power Delivery*, 28(3):1682–1690, July 2013.

[73] K. R. Padiyar. *POWER SYSTEM DYNAMICS: Stability and Control*. BS Publications, 2nd edition edition, 2008.

[74] S. Wolfram. *A New Kind of Science*. Wolfram Media, 2002.

[75] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Elsevier Science, 2nd edition edition, 2004.

[76] C. J. Geyer. *Introduction to Markov Chain Monte Carlo, in Handbook of Markov Chain Monte Carlo: Methods and Applications*. CRC Press, 1nd edition edition, 2010.

[77] J. K. Kruschke. *Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan*. Academic Press, 2nd edition edition, 2014.

[78] C. Robert and G. Casella. A short history of markov chain monte carlo: Subjective recollections from incomplete data. *Statistical Science*, 0:1–14, 2011.

[79] D. B. Hitchcock. A history of the metropolis-hastings algorithm. *The American Statistician*, 57:254–257, 2003.

[80] J. Castro B. Balle and R. Gavalda. Learning probabilistic automata: A study in state distinguishability. *Theoretical Computer Science*, 473:46–60, 2013.

[81] Y. Esposito P. Duponta, F. Denis. Links between probabilistic automata and hidden markov models: Probability distributions, learning models and induction algorithms. *Pattern Recognition*, 38:1349–1371, 2005.

[82] C. Unsal. *Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles*. PhD thesis, Virginia Polytechnic Institute and State University, 1997.

[83] K. N. Wexler. *An Automaton Analysis of The Learning of Miniature System of Japanese*. PhD thesis, Stanford University, 1970.

[84] Robert U. Ayres. Technological forecasting and long-range planning. 1969.

[85] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[86] D. Ron. *Automata Learning and Its Applications*. PhD thesis, Hebrew University, 1995.

[87] K.S. Narendra and M.A.L. Thathachar. Learning automata a survey. *IEEE Transactions in Systems, Man and Cybernetics*, SMC-4:4, 1974.

[88] R. W. Mclaren. A stochastic automaton model for the synthesis of learning systems. *IEEE Transactions on Systems Science and Cybernetics*, 2(2):109–114, Dec 1966.

[89] Andrés Takach, Wayne Wolf, and Miriam Leeser. An automaton model for scheduling constraints in synchronous machines. *IEEE Trans. Comput.*, 44(1):1–12, January 1995.

[90] Teruo Higashino, Akio Nakata, Kenichi Taniguchi, and Ana R. Cavalli. *Generating Test Cases for a Timed I/O Automaton Model*, pages 197–214. Springer US, Boston, MA, 1999.

[91] Irun R. Cohen and Henri Atlan. Network regulation of autoimmunity: An automaton model. *Journal of Autoimmunity*, 2(5):613 – 625, 1989.

[92] D. Dhar. Self-organized critical state of sandpile automaton models. *Phys. Rev. Lett.*, 64:1613–1616, Apr 1990.

[93] Ravi Mirchandaney and John A. Stankovic. Using stochastic learning automata for job scheduling in distributed processing systems. *Journal of Parallel and Distributed Computing*, 3(4):527 – 552, 1986.

[94] Nils Klarlund. *Mona & Fido: The logic-automaton connection in practice*, pages 311–326. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[95] Karel Culik and Jarkko Kari. Image compression using weighted finite automata. *Computers & Graphics*, 17(3):305 – 313, 1993.

[96] Borja Balle, Jorge Castro, and Ricard Gavald. Learning probabilistic automata: A study in state distinguishability. *Theoretical Computer Science*, 473:46 – 60, 2013.

[97] Colin de la Higuera and Jose Oncina. *Learning Stochastic Finite Automata*, pages 175–186. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[98] L. Naumov and A. Shalyto. Automata theory for multi-agent systems implementation. In *Integration of Knowledge Intensive Multi-Agent Systems, 2003. International Conference on*, pages 65–70, Sept 2003.

[99] Kaan Ozbay, Aleek Datta, and Pushkin Kachroo. Application of stochastic learning automata for modeling departure time and route choice behavior. *Transportation Research Record: Journal of the Transportation Research Board*, 1807:154–162, 2002.

[100] Pinaki Chakraborty, P. C. Saxena, and C. P. Katti. Fifty years of automata simulation: A review. *ACM Inroads*, 2(4):59–70, December 2011.

[101] Kimmo Koskenniemi. A general computational model for word-form recognition and production. In *Proceedings of the 10th International Conference on Computational Linguistics*, COLING '84, pages 178–181, Stroudsburg, PA, USA, 1984. Association for Computational Linguistics.

[102] Peter Linz. *An Introduction to Formal Language and Automata.* Jones and Bartlett Publishers, Inc., USA, 2006.

[103] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, April 1994.

[104] Mehryar Mohri, Fernando Pereira, and Michael Riley. The design principles of a weighted finite-state transducer library. *Theoretical Computer Science*, 231(1):17 – 32, 2000.

[105] Yongming Li. Finite automata theory with membership values in lattices. *Information Sciences*, 181(5):1003 – 1017, 2011.

[106] Ch.-A. Gandin and M. Rappaz. A coupled finite element-cellular automaton model for the prediction of dendritic grain structures in solidification processes. *Acta Metallurgica et Materialia*, 42(7):2233 – 2246, 1994.

[107] David Pfau, Nicholas Bartlett, and Frank Wood. Probabilistic deterministic infinite automata. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1930–1938. Curran Associates, Inc., 2010.

[108] A. Subhadra. Automaton-an abstract computing devise & its recent trends. *International Journal of Application or Innovation in Engineering & Management*, 5:128–136, 2016.

[109] Dana Ron, Yoram Singer, and Naftali Tishby. On the learnability and usage of acyclic probabilistic finite automata. *Journal of Computer and System Sciences*, 56(2):133 – 152, 1998.

[110] Michael Sipser. *Introduction to the Theory of Computation.* International Thomson Publishing, 1st edition, 1996.