A STATISTICAL MECHANICS APPROACH FOR AN EFFECTIVE, SCALABLE, AND RELIABLE DISTRIBUTED LOAD BALANCING SCHEME FOR GRID NETWORKS

OSAMA ABU-RAHMEH

A thesis submitted in partial fulfilment of the
requirements of Liverpool John Moores University
for the degree of Doctor of Philosophy

School of Engineering,
Liverpool John Moores University

March 2009

"Allah will exalt in degree those of you who believe and those who have been granted knowledge. And Allah is well-acquainted with what you do"

Holy Quran

Chapter 58. Verse 11

The Prophet Mohammad, peace be upon him, Said "Whoever takes a path in search for knowledge, Allah will facilitate for him a path to Paradise"

Prophet Teachings

Dedicated to:

My beloved parents and Wife

A Statistical Mechanics Approach for an Effective, Scalable, and Reliable Distributed Load Balancing Scheme for Grid Networks

By

Osama Abu-Rahmeh

Abstract

The advances in computer and networking technologies over the past decades produced new type of collaborative computing environment called Grid Networks. Grid network is a parallel and distributed computing network system that possesses the ability to achieve a higher computing throughput by taking advantage of many computing resources available in the network. To achieve a scalable and reliable Grid network system, the workload needs to be efficiently distributed among the resources accessible on the network.

A novel distributed algorithm based on statistical mechanics that provides an efficient load-balancing paradigm without any centralised monitoring is proposed here. The resulting load-balancer would be integrated into Grid network to increase its efficiency and resources utilisation. This distributed and scalable load-balancing framework is conducted using the biased random sampling (BRS) algorithm.

In this thesis, a novel statistical mechanics approach that gives a distributed load-balancing scheme by generating almost regular networks is proposed. The generated network system is self-organised and depends only on local information for load distribution and resource discovery. The in-degree of each node refers to its free resources, and job assignment and resource updating processes required for load balancing are accomplished by using random sampling (RS). An analytical solution for the stationary degree distributions has been derived that confirms that the edge distribution of the proposed network system is compatible with ER random networks. Therefore, the generated network system can provide an effective load-balancing paradigm for the distributed resources accessible on large-scale network

systems. Furthermore, it has been demonstrated that introducing a geographic awareness factor in the random walk sampling can reduce the effects of communication latency in the Grid network environment. Theoretical and simulation results prove that the proposed BRS load-balancing scheme provides an effective, scalable, and reliable distributed load-balancing scheme for the distributed resources available on Grid networks.

Acknowledgement

It has been an honour and a privilege to be associated with ***Dr. Princy Johnson*** and ***Prof. A. Taleb-bendiab*** for carrying out the research work towards my PhD degree. Their professional guidance, invaluable advice, patience, constant support and encouragement have made this thesis possible. Therefore I would like to take this opportunity to express my gratitude and sincere thanks to them.

Some parts of this work wouldn't have been possible without the active contribution of ***Sebastian Lehmann***, and for that, I am very grateful.

I am also grateful to ***Adoum Lamine*** and all other colleagues for their advice, interesting discussions, encouragement, and helpful suggestions and for the good time I spent with them.

A special thank to ***my parents*** and ***my lovely wife***, ***Huda***, for their endless support and motivation throughout my Ph.D. studies. Without their constant assurance and assistance, completion of this research work would have not been possible. And as a sign of my love, gratitude and affection I dedicate this thesis to them.

Thank you all,

Osama Abu-Rahmeh

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

API	Application Programming Interface
BA	Barabási–Albert Model
BOINC	Berkeley Open Infrastructure for Network Computing Project
BRS	Biased Random Sampling
BS	Base Station
CPU	Central Processing Unit
ER	Erdős–Rényi Model
GPU	Gnutella Processing Unit
GridFTP	Grid File Transfer Protocol
IBM	International Business Machine
IT	Information Technology
IP	Internet Protocol
LDAP	Lightweight Directory Access Protocol
lwIP	Lightweight TCP/IP
Mbps	Mega Bits per Second
OGSA	Open Grid Service Architecture
P2P	Peer-to-Peer Architecture
PingER	Ping End-to-end Reporting Project
QoS	Quality of Service
RS	Random Sampling
RTL	Round Trip Latency
RTT	Round Trip Delay Time
SF	Scale-Free Networks Model

SOAP	Simple Object Access Protocol
SW	Small-World Networks
UDP	User Datagram Protocol
WS	Watts and Strogatz Model
WSNs	Wireless Sensor Networks
XML	Extensible Mark-up Language

N	Number of nodes in the network (i.e. network size)
p	The probability that a node will connect to other node in the graph
k	The degree of the node which is the number of edges the node has to other nodes
p_k	The probability that a node has k edges
E	The average number of edges in the network
k_{in}	The incoming degree which is the number of links that point to a node
k_{out}	The outgoing degree which is the number of links that start from
L	Total number of edges in the graph
$\overline{k}$	The node's average degree in the network
$P(k)$	The degree distribution of the network which gives the probability that a randomly selected node has k edges
λ	The exponent of the power-law degree distribution
i	A node in the network
k_i	Number of edges that connect node i to k_i other nodes
C_i	The clustering coefficient of node i
C	The clustering coefficient of the whole network
E_i	The number of edges that exist between the k_i nodes

d	The diameter of a graph which is the maximum distance between any pair of its nodes
l_{rand}	Average path length of a random graph
C_{N-1}^{k}	The probability space in which k edges are chosen from the total number of edges $N-1$
X(k)	The probability to add a link to a certain node
p	The probability that a node will be connected to another node
α	An exponent describes the way in which probability grows with degree
T_{sender}	Threshold value of load index for a node to become a sender
$T_{receiver}$	Threshold value of load index for a node to become a receiver
D	The average number of deleted edges in the network
M	The maximum number of edges a node can have
R_k	The rate that the in-degree of a specific node with k edges will decrease
S_k	The rate that the in-degree of a specific node with k edges will increase
A	The jobs transition matrix
I	The identity matrix
V	The distribution vector (or transition probability)
Cp	Connection probability for the nodes in the network
$\mathrm{Var}(k)$	The in-degree variance of the network
L_i	The workload offered by each node i in the network
H_i	The control packets used for handshaking protocols
T_i^{L}	The time spent to send the workload packets
T_i^{H}	The time spent to send the control packets

BW_{Total}	The total bandwidth consumed by the entire network
BW_i	The bandwidth consumed by node i in the network
J_i	The number of completed jobs by node i in the network
$\bar{\nu}$	The expected job arrival rate
$P(\nu)$	The job arrival rate
ν_{max}	The maximum number of jobs that can arrive at any time in the network
$\bar{\beta}$	The average job size
β	The size of the job
T	Simulation time slot
$T_{Simulation}$	Total simulation time
$Throughput_{Total}$	The total throughput attained in the network
j	The number of simultaneous random sampling sent by a node
ℓ_{RS}	The random sampling length

Chapter 1

INTRODUCTION

Chapter 1

INTRODUCTION

1.1 Introduction

Improvements in computer and networking technologies have produced a dramatic increase in communication and computer capabilities. During the past decades, numerous methods have been developed to maximise the use of networked computers for large-scale computing, and several protocols have been developed to efficiently utilise resources within a distributed computing system.

Large-scale computing networks enabled a new kind of application to emerge where the network is populated with services and devices with different Quality of Service (QoS) requirements. Therefore, next-generation systems will require predictable behaviour in areas such as throughput, load balancing, routing, scalability, dependability, and security. There is a need for more flexible network systems that can adapt to dynamic changes in application requirements and environmental conditions, and its flexibility should not be at the expense of reliability or security. Moreover, the spread of the Internet as well as the availability of powerful computers and high-speed network technologies are rapidly changing the computing landscape and society. All these developments in technology have led to the possibility of using wide-area distributed computers for solving large-scale problems.

Large-scale computing systems can provide higher throughput computing by taking

advantage of many networked computers that create a virtual computing architecture to distribute process execution between the computers in the network. An example of such network system is the Grid Network (Foster and Kesselman, 1999).

Grid Network is a parallel and distributed computing network system that utilises the available resources in the network to solve large-scale computation problem. With Grid Network, organisations can optimise computing and data resources, pool them for large capacity workloads, share them across networks and enable collaboration. Therefore, Grid Networks possess the ability to achieve a higher throughput computing by taking advantage of many computing resources available in the network. However, to achieve a scalable and reliable Grid Network system, the workload needs to be efficiently distributed among the resources accessible on the network.

In this thesis, a novel distributed algorithm based on statistical mechanics that provides a load-balancing paradigm without any centralised monitoring, is proposed. The resulting load-balancer would be integrated in Grid Network to increase its efficiency and resources utilisation. The proposed load-balancing framework is implemented using biased random sampling (BRS) technique. The generated network system provides an effective, scalable, and reliable non-centralised load-balancing scheme for the distributed resources available on Grid Networks.

1.2 Research Motivation

Grid Network is an active research field (NGG2, 2004). There are several challenges that need to be addressed in order to design and optimise Grid Networks. Research is

ongoing in a number of areas including resource allocation, load distribution, mathematical models, performance measurements, routing mechanisms, and fault tolerance techniques (Kephart and Chess, 2003; Foster, 2002; NGG2, 2004; Favarim et al., 2003; Jennings, 2000).

In order to meet these challenges, appropriate models and techniques are needed for understanding, controlling, and designing the emergent behaviour in large network systems, such as Internet and Grid Networks. Fundamental mathematical work is needed to understand how the properties of self-configuration, self-optimisation, self-maintenance, and robustness arise from or depend on the behaviours and goals of individual elements. In addition, it can be used to investigate the pattern and type of interactions among them, and the external influences or demands on the system.

Recently, a new area called Complex Networks Theory emerged (Scharnhorst, 2003). Complex Networks theory is the field where the structural and dynamic properties of networks are analysed. Statistical models of large networked systems will let systems detect or predict overall performance problems from the stream of data from individual devices.

There has been a large improvement in the field of Complex Networks due to combining ideas and analytical tools from statistical mechanics. These analytical tools have led to develop a number of protocols and models for Complex Networks that result in predictable properties. Networks have various dynamical processes, and their topology determines their dynamical features.

Many models and concepts have been developed to describe and analyse the Internet and the web (Adamic, 1999; Albert, et al., 1999; Dorogovtsev, et al., 2000; Faloutsos et al., 1999; Huberman, 2001; Huberman and Adamic, 1999; Tadic, 2001; Vázquez et al., 2002; Krapivsky and Redner, 2002; Barabási, 2001).

However, analysing and studying Grid Networks using Complex Networks theory is still new and a challenging research problem. Besides, the current models do not allow the networks to grow up and increase its size, but they exist in a stationary state that supports different network topologies (Slanina and Kotrla, 2000, Slanina and Kotrla, 1999, Newman, et al., 2001). Thus, it is still a challenge to design new models that based on selection or optimisation mechanisms would produce topologies similar to those seen in the real world.

1.3 Research Objectives

Grid Network consist of millions of interconnected nodes, and with its huge number of distributed resources, it is likely that some nodes are heavily loaded while others are lightly loaded or even idle. To achieve maximum use of these large systems, it is desired that the workload be distributed among all the nodes so that resource utilisation is efficient and maximum performance is achieved. A load distribution scheme must decide where and when a given task should be executed in order to increase performance. Therefore, implementing an effective load-balancing technique that distributes the load among the available nodes in the network can improve overall system performance. Thus, when one node is overwhelmed by work, it can make use of unused computing power in the network. Therefore, integrating an effective load-balancing paradigm for an efficient load distribution and resource

discovery will have a significant influence in implementing the self-configuring and self-optimising characteristics of Grid Networks.

The principal objectives of this research can be summarised as follows:

1. *To develop a statistical mechanics network system based on Complex Networks theory that would provide an efficient load-balancing scheme for Grid Networks.*
2. *To analyse and derive the stationary degree distribution for load distribution in the proposed network system.*
3. *To develop an efficient, reliable, and scalable load-balancing protocol suitable for use in large-scale networks based on a biased random sampling (BRS) scheme.*
4. *To determine the proper length of random sampling required to balance the network.*
5. *To evaluate the performance of the proposed load-balancing scheme through Simulations and analyse the results. The random sampling will be biased in various ways to select the nodes, such as:*
    a. *Geographical position and communication latency between the nodes.*
    b. *Computational power (such as CPU) available for each node.*
    c. *Number of resources available for each node.*
6. *To extend the network simulations to take into consideration the heterogeneous nature of nodes capability and job size to predict the performance of the algorithm in large-scale networks.*
7. *To evaluate the efficiency, scalability, and reliability of the proposed load balancing mechanism in various situations.*

1.4 Contributions

The field of load balancing is an active research area and many techniques and problem formulations have been used. However, implementing an efficient load balancing mechanism is a challenging research area. There is a need to consider the cases where services are time-bound, or require certain QoS requirements, or when jobs' execution process depends on the outcome of other jobs.

To address these issues, a *novel* statistical mechanics system that provides a distributed load balancing mechanism is proposed in this thesis. This novel network system leads to a *decentralised, self-organised,* and *scalable* network that depends only on local information for load distribution and resource update. Since the resulting network system is based on random graphs, there is a higher possibility to find a connected path between almost any two nodes in the system. Therefore, it will be more *resilient* to random errors.

A *novel* load-balancing protocol based on random sampling is developed that is suitable for use in both *homogeneous* and *heterogeneous* large-scale networks. A stationary solution for load distribution which uses random sampling technique to distribute and balance the load in the network has been derived. Analytical and simulation results have been used to evaluate and validate the proposed load-balancing scheme.

Though similar techniques exist for load balancing, the proposed scheme has the advantage of *dynamically reshaping* the network structure to efficiently distribute the load. Moreover, this load-balancing paradigm will not require any monitoring

mechanisms since it is intrinsic in the network structure, and the random sampling algorithm which will be used for node selection will depend on local information about the free resources available for each node.

The proposed load-balancing scheme uses a decentralised and scalable balancing algorithm. While other decentralised load-balancing algorithms have been proposed in the literature, performance and scalability analysis for the algorithms which promise nearly optimal performance as the number of nodes becomes very large have been lacking. Scalability is capability of a load-balancing algorithm to work efficiently when applied to a large number of nodes. It has been demonstrated that under ideal conditions, the network structure converges to a random graph that is at least as regular and balanced as Erdős-Rényi (ER) random graphs (Erdös and Rényi, 1960).

In addition, the random sampling technique is improved by *biasing* the sampling walk toward specific nodes such as unvisited nodes or nodes with certain properties instead of choosing them uniformly at random. Hence, the nodes' selection criteria will be based on a predefined criterion rather than selecting only the last node in the walk. Moreover, the number of sampling steps (or sampling length) will be *limited to a specific length* of order $O(\log N)$, where N is the number of nodes in the network.

Another essential feature of the Grid Network is that the resources accessible in the network are distributed geographically. However, one of the fundamental challenges to run Grid applications across geographically distributed computational resources is to overcome the effects of the latency between them. Although high performance

clusters and supercomputers can deliver data to applications with a latency of few microseconds, latency across the wide area networks is measured typically in milliseconds. Thus, reducing the effects of communication latency is critical for achieving good performance with Grid applications that involve significant amount of communication.

Therefore, a novel biased random sampling algorithm is proposed that *optimises* the *communication latency* in Grid Networks and thus enabling the network to achieve load balancing which is scalable and reliable.

1.5 Thesis Structure

The thesis is divided into eight Chapters. Chapter 1 is an introductory Chapter that outlines load balancing and its applications and the contributions of the thesis. Details of the scope of this research are also given here. In Chapter 2, the literature review related to Grid Networks is presented and a detailed review of the existing of techniques of load-balancing is given. Chapter 3 gives a review of Complex Networks Theory and discusses the concepts and standards of this area. Chapter 4 presents the modelling techniques used to describe the networks as random networks. Chapter 5 proposes a novel biased random sampling (BRS) load balancing scheme for Grid Networks. the analytical solution for the stationary degree distribution of the network is presented here. Moreover, simulation description and implementation are summarised in this chapter. Performance evaluation of the proposed algorithm and the research results are reported in Chapter 6. In addition, the new latency optimising biased random sampling algorithm is presented here. In Chapter 7, conclusions are drawn from the research described in this thesis.

Furthermore, suggestions for further work are outlined in this Chapter. Finally, references are listed in Chapter 8.

Chapter 2

LITERATURE REVIEW

Chapter 2

LITERATURE REVIEW

The spread of the Internet as well as the availability of powerful computers and high-speed network technologies at low-cost commodity components is rapidly changing the computing landscape and society. These technology opportunities have led to the possibility of using wide-area distributed computers for solving large-scale problems, leading to what is popularly known as Grid Network (Foster and Kesselman, 1999).

2.1 Grid Network Technology

Grid Network is a computing system that provides the ability to perform higher throughput computing by taking advantage of many networked computers to model a virtual computer architecture that is able to distribute process execution across a parallel infrastructure. Grid Network uses the resources of many separate computers connected by a network to solve large-scale computation problems. It is a type of parallel and distributed system that enables the sharing, selecting, and assembling of geographically distributed resources dynamically at runtime, depending on their capability, availability, cost, performance, and users' quality-of-service requirements.

Grid Networks represent the idea of a promising infrastructure that is focused on networking together heterogeneous, multiple regional and national computing systems. Grid Networks enable software applications to integrate instruments,

displays, and computational and information resources that are managed by various organisations in widespread locations. It provides an abstraction for resource sharing and collaboration across multiple administrative domains. The concept resource covers a wide range of terms including physical resources (computation, communication, storage), informational resources (databases, archives, instruments), individuals (people and the expertise they represent), capabilities (software packages, brokering and scheduling services) and frameworks for access and control of these resources such as OGSA and Semantic Web (Foster, 2002).

A Grid Network is a services-oriented architecture that contains heterogeneous systems and involves distributed computing over the Internet or any private network via open standards. It enables the virtualization of distributed computing and data resources such as processing, network bandwidth and storage capacity, to create a single system image, granting users and applications seamless access to vast IT capabilities. Just as an Internet, user views a unified instance of content via the Web; a Grid Network user essentially sees a single, large virtual computer.

Therefore, Grid Networks will be a virtual infrastructure with specific computational semantics. It performs computation, solves problems, or provides service to a single or million clients. The Grid Network may consist of millions of interconnected nodes. A Grid Network node is an atomic unit forming an abstraction over resources, entailing what is hidden by the interfaces it provides. Nodes may provide new services, functions, or even new concepts that are unknown to clients, and they can be organised, at runtime, into a group in order to provide functionality and behaviours that none of its individual members has. The self-organising capabilities

of nodes aim at establishing higher robustness and lower costs for systems management. These capabilities are provided through a small, common set of facilities, such as highly scalable protocols for communication and membership management.

At its core, Grid Network is based on an open standards and protocols (e.g. Globus, OGSA (Foster, et. al, 2002), Condor-G (Frey, 2001), GridFTP (Taylor, 2005) and Grid Resource Information Protocol (Czajkowski, 2001)) that facilitate communication across heterogeneous, geographically dispersed environments. With such infrastructure, organizations can optimize computing and data resources, pool them for large capacity workloads, and share them across networks, thus enabling collaboration.

The Globus Project launched a research and development program aimed at creating a toolkit based on the Open Grid Service Architecture (OGSA) (Foster, et. al, 2002) that defines standard mechanisms for creating, naming, and discovering services and specifies various protocols to support accessing services. OGSA specifies an open standard for Grid Network protocols and interfaces, supplies Web services, and it is designed to enable large-scale cooperation and access to applications through the Internet.

2.1.1 Grid Network and Autonomic Computing

Autonomic Computing (Ganek and Corbi, 2003) initiative, which is focused on making software and servers that are self-optimising, self-configuring, self-protecting and self-healing, is closely related to the Grid Network concept. That is,

the principles of Autonomic Computing are similar to Grid Network principles which include the development of intelligent, open systems that are capable of adapting to varying circumstances and preparing resources to efficiently handle the workloads placed upon them. Therefore, autonomic technologies will help companies to manage their Grid Networks more easily and cost-effectively. And since the spread of the Grid Network can expand the domain of computing across many systems, a successful Grid Network system will require autonomic functionality.

Like other elements of e-business on demand (the Internet, the Web, Linux, etc.), Grid Network is more powerful with open standards. Work efforts on Grid Network and Autonomic Computing are creating important architectural models and new open industry standards which will help the IT industry to make progress toward more self-managing systems.

2.1.2 Wireless Grid Networks

The popularity of wireless devices, such as laptop computers, mobile phones, personal digital assistants, digital cameras, and so on, is rapidly increasing. These devices can be connected to wireless networks of increasing bandwidth, and software development kits are available that can be used by third parties to develop applications. Moreover, they can be used to access personal and public information, to store personal information, and to communicate with others. Therefore, what we expect from those devices is the right information displayed at the right time on the right device in the right format, and with the right level of intrusiveness upon the user's current tasks. This requires context-awareness: the location of the device, the

profile of the user(s), their current schedule and tasks, the capabilities of the interface, aspects of security and trust. Furthermore, they can also be used to capture information as tools to support the users in a creative process of collecting and working with information and knowledge (Globus, 2003).

It is obvious that mobile devices will make significant demands on my information processing capabilities and computational infrastructure. It requires the automatic generation of information from multiple sources and customised for delivery to the user's device. Furthermore, it may require computation for information search and generation for the purposes of modelling and prediction, which in turn may inform the dynamic behaviour of the devices. In addition, it may require an on-demand and timely presentation of information, requiring dynamic composition and negotiation of services. This creates challenges for negotiation, coordination, and scheduling.

To achieve this, Grid Network infrastructures are required. They require the assembly of data and computational resources to meet application requirements, they involve the creation and support of virtual organizations, and they need an infrastructure, which provides interoperability but also provides security and appropriate models for service negotiation and charging. They also require a user-centric perspective, working with context and user information, and new qualities of Grid Networks service to address the dynamic aspects such as timeliness and change. In addition, it would be desirable if mobile devices can co-operate independently from the central station in a distributed mobile ad hoc network to perform a set of services. This would be necessary in cases where the central station has been damaged or is temporarily unavailable. This is also necessary in cases where small

teams have to operate independently for some time, and local data can be cached, avoiding having overhead to communication links and computational capacity of the central station. This aspect also refers to the ability of the mobile staff to have access to remote information and databases distributed over the world.

However, such applications on these types of devices introduce challenging problems. Devices face temporary and unannounced loss of network connectivity when they move. They need to discover other hosts in an ad hoc manner, and they are likely to have limited resources, such as low power battery, slow CPU, and little memory. Furthermore, they are required to react to frequent changes in the environment such as changes of location or context conditions, and variability of network bandwidth (Coulouris, et al., 2001).

2.2 Grid Networks Architectures

Grid Networks show a variation in purpose and architecture, however, there are few common techniques where communication between nodes occurs on a Grid Network. The two most common protocols are the *master-slave* and *peer-to-peer* (P2P) *architectures*.

2.2.1 The Master-Slave Architecture

The master-slave (or client-server) architecture is the least complicated and most common communication setup for Grid Network. It involves the use of a master node and one or more slave nodes. The master node sends control signals to the slave nodes and the client executes or responds to those commands. Many of the current Grid Network computers use the master-slave architecture. The master-slave

architecture uses a simple and direct communication between the nodes and the server, and since the instructions and tasks come from a centralised server, it can be easily verified and validated. Moreover, it is easy to return results as they are collected by the server controlling the nodes. However, reliability and scalability are a major challenge on a centralised architecture which makes it more liable to single point of failure. In addition, the server needs to support the clients with a lot of resources (such as: bandwidth, memory, CPU, disk) which will make it a bottleneck on the network with a large number of nodes (clients).

Examples of master-slave based architecture include the BOINC (Anderson, D., 2004) and the Globus (Globus Alliance, 2005) toolkit. BOINC is an open source framework built to power scientific projects; it is developed by a group of volunteers and participants from the University of California, Berkeley. BOINC forms the basis of several well known Grid Network computing projects including the SETI@Home project, which is a Grid Network computing application designed for a "scientific experiment that uses Internet-connected computers in the Search for Extraterrestrial Intelligence" (SETI Project 1999-2005). It is also used to power more generic community Grid Network initiatives, such as the world community Grid Network (IBM Corporation 2005b).

The Globus Toolkit is another sophisticated open source Grid Network framework, which has many years of research and development behind it, and a number of developers from various universities around the world. It's targeted to a number of open and commercial research projects based around e-science and e-business applications (Globus Alliance 2005).

Both BOINC and the Globus Toolkit allow a developer or group of developers to access their application programming interface (API). This API exhibits functionality such as: message routing, XML Simple Object Access Protocol (SOAP), security control and definition, and retrieval of statistics, on top of the core features of other Grid Network.

2.2.2 The Peer-to-Peer Architecture

The peer-to-peer (P2P) architecture has been a more recent development as they involve the use of peer-to-peer protocols to manage the underlying communication channels. Peer-to-peer protocols have a large number of applications, and most of these protocols were originally designed to address the problem of decentralised message/content delivery. As these protocols have become more refined, standardised and understood by the developer community, the benefits of their application have become better understood.

Although P2P's utility has not yet been fully realised outside of the content distribution space, a few new and promising frameworks have begun the implementation of a Grid Network framework based on P2P. (Mengotti, et al, 2002) describe one such framework based on the Gnutella P2P protocol, called the Gnutella Processing Unit (GPU). The GPU framework uses P2P communication as a way to send messages to client nodes (GPU team 2004-2005).

The key difference between P2P and master-slave architectures lies in the control of resources and control delegation (Mengotti, 2004). In a P2P network, nodes join and leave the network without ever being connected to a master server; this is because all

nodes act as masters and/or slaves. Control messages and statistics are passed from one node to another through peering. This means that the Grid Network works cooperatively and the throughput of the Grid Network is determined by the number of connections that a client node has to its peers.

The advantage of the P2P architecture is it is de-centralised and there is no need for dedicated servers since each node can act as both server and client. Thus, the larger the size of the network, the better is its reliability (Mengotti, et al, 2002). Moreover, since jobs can be given to the next available peer if a node goes offline, this makes the network reliable and more resilient to node failures.

However, the acquisition of statistics is difficult as messages may need to be routed over several nodes. Moreover, processing overhead is involved in validating and verifying the authenticity of control signals or commands. Also, a dynamic technique is required for resource discovery and their availability. Some of the major technical challenges in P2P Grid Network computing include the construction of security measures needed to stop viral infections on the Grid Network and providing a trust model for inter-node communication. That is, a node must be able to verify the integrity and authenticity of a message sent by a peer.

2.3 Load-balancing: A Background

The development of Grid Networks requires solutions to a number of technical challenges (e.g., load balancing, routing, flow control, resource discovery and state maintenance, source and channel coding, power control, modem design, hardware,

etc.). Among all these, the load balancing and state maintenance problems are of particular interest.

Load-balancing is the process of distributing the computation workload across multiple machines/processors as evenly as possible with an objective to improve the overall system performance (Peixoto, 1996). The load distribution scheme must decide where and when a given job should be executed in order to increase performance. Application-level load-balancing algorithms concentrate on minimising the execution time of the jobs. System-level load-balancing and distributed scheduling focuses on maximising system throughput or the overall utilisation rate of the network.

The objectives of load-balancing algorithms may vary and influence the design of the load distribution strategy (Riedl, and Richter, 1996). The typical objectives of load-balancing algorithms are:

- To balance the distribution of workload.
- To minimise the jobs average response time.
- To maximise the network throughput.
- To predict task response times in real-time systems.
- To increase the reliability and fault-tolerance of the system.

For example, it has been shown that in a heterogeneous system, the distribution of loads across the heavily loaded nodes into lightly loaded nodes would improve the performance of all nodes in the system (Zhou, 1988).

The performance of the load distribution mechanism may be viewed from either the workload or the system point of view. When considering performance from the process point of view, the metric used is often the individual job's completion time response. When viewing the system as a whole, it is desired that all jobs are being treated fairly and that the system throughput is improved. The system point of view of improving resource utilisation is compatible with the desire for maximum system throughput. Though, there is an inherent conflict in trying to optimise both response and throughput. In a parallel system, it is often required to run a single application composed of several tasks and performance is measured as the completion time of this application.

When evaluating a load-balancing mechanism there are two properties which must be considered (Casavant and Kuhl, 1988):

- ***Performance***: the satisfaction of the tasks with how well the mechanism manages the resources in question.
- ***Efficiency***: the satisfaction of the tasks in terms of how costly it is to access the load distribution mechanism itself.

In other words, the tasks and applications would like to quickly access the machines or processors, but do not want to be delayed by overhead problems associated with using the management function itself. The desirable system behaviour is that which has the highest level of performance possible while incurring the least overhead in doing it.

It is possible to identify some requirements which proved to be important for a general purpose load balancing strategy (Kunz, 1991):

- No apriori knowledge about incoming tasks requirements.
- No assumptions about the underlying network (topology, homogeneity, size, etc).
- Dynamic, physically distributed, and cooperative decision making.
- Minimisation of average/worst response time of tasks as performance criteria.

2.4 Load-Balancing Components

Generally, a load-balancing scheme consists of four components: information gathering, data transfer, decision making, and data relocation processes (Shivaratri, 1992).

2.4.1 The Information Gathering Process

The information gathering process decides when information about the states of other nodes is to be collected, from which nodes and what information is collected. It collects the information of load distribution state and detects if there is differences in load distribution. It is also responsible for the broadcasting of each node's load information.

A key issue is to identify a suitable load index which is able to describe the current load of a node. A number of load indices have been used in the literature: CPU queue length, CPU utilization, normalised response time, I/O queue length, memory utilization, context-switch rate, and system-call rate (Ferrari and Zhou, 1987, Kunz, 1991). It has been observed that a task at a node is likely to demand services from a

number of resources, so it might be important to define load not only as a single resource in a node, but as a collection of resources.

2.4.2 The Transfer Process

The transfer process decides if there is a need to initiate load balancing across the system (Gustafson, 1988, Becker and Waldmann, 1994). Using load information, the transfer process determines when a node becomes appropriate to act as a sender (transfer a job to another node) or as a receiver (retrieve a job from another node).

Transfer processes may be either based on thresholds or relative transfer processes. Threshold policies decide that a node is a sender if its load index exceeds a threshold T_{sender} or a receiver if it falls below a threshold $T_{receiver}$ (Shivaratri, 1992). The choice of these thresholds is fundamental to the performance of the algorithm. Clearly the best threshold values depend on the system load and the task transfer cost.

Relative transfer processes consider the difference between the load of a node and the loads of the other nodes in the network. Nodes might be considered able to participate in a transfer if their loads differ by more than some threshold function. They might then transfer some fixed number of tasks or a fraction of the load difference (Casavant and Kuhl, 1988, Kropf, 1996, Lüling, et. al, 1991, Monien, 1996, Scheurer, et. al, 1995, Xu, et. al, 1995).

Additionally, the transfer process may be either periodic or event-triggered. The algorithm may periodically check if the node's state qualifies itself as a candidate for a task transfer or not. However, the great majority of the algorithms proposed in the

literature are event-triggered. If the state of the node or one of its neighbours changes, then, a task transfer may be possible. The state of a node may change because a task has ended or because a new task has been initiated.

2.4.3 The Decision Making Process

The decision making process calculates the optimal data distribution. Once a node decides to participate in a task transfer as a sender, a decision making process must select the task(s) to be transferred. The decision making process should consider several factors:

- The overhead incurred in transferring the task should be minimised.
- The execution time of the transferred task should be enough to justify the cost of the transfer.
- The number of location-dependent resources needed by the selected task should be minimal. These resources include specific data, I/O devices (display, keyboard, disks, etc).

2.4.4 The Data Relocation Process

The data relocation process selects a suitable transfer node using information about the nodes' states for a job transfer transaction (Kalé, 1988, Kumar, 1994). In other words, it locates the nodes to/from which a node can send/receive workload to improve overall system performance. It transfers the load from over-loaded machines to unloaded ones. Some algorithms try to find the best node among the same network cluster, while others just look for an acceptable node from the entire network. For example, in a random relocation algorithm, the destination node is randomly selected among all the nodes in the system. The random relocation scheme

chooses a random node without using any information about the target's state (Kumar, et. al, 1992, Shin and Chang, 1995). The random scheme yields substantial performance improvement over the non-load sharing case. The degree of performance improvement is surprisingly high for such a simple algorithm. The authors also claim that this is a very scalable algorithm since it does not collect any system state information.

Some relocation processes use probabilistic schemes instead of state-dependent ones. Probabilistic schemes distribute tasks according to a set of predefined rules. Studies have shown that state-dependent relocation schemes consistently outperform their probabilistic counterparts (Stankovic, 1985).

Data relocation processes can be generally classified as *sender-initiated* (Eager, et. al, 1986, Zhou, 1988), *receiver-initiated* (Eager, et. al, 1986, Lin and Raghavendra, 1992), or *symmetrically-initiated* (Shivaratri, et. al, 1992, Willebeek-LeMair and Reeves, 1993, Feng, et. al, 2000,). Sender-initiated policies are those where heavily-loaded nodes search for lightly-loaded nodes while receiver-initiated policies are those where lightly-loaded nodes search for suitable senders. Symmetrically-initiated policies combine the advantages of these two by requiring both senders and receivers to look for appropriate partners.

The relocation process may deal with some restrictions when looking for a destination node. These restrictions may include resource requirements, tasks precedence, and data dependencies.

2.5 Load-Balancing Scalability and Stability

A very important property of load-balancing policies is scalability, which implies that an algorithm should be independent of system size and physical resource characteristics such as communication bandwidth and processor speed. Scalability analysis of a parallel algorithm and architecture combination is very useful to predict the performance of a system when there are changes on the number of nodes or on the size of the problem instance being solved.

Stability is the ability of the algorithm to detect when the effects of further actions will not improve the system state (Peixoto, 1996). A stable algorithm will return the system to a state of equilibrium after a disruption from this equilibrium. In the context of load distribution, a disruption is the arrival or removal of tasks, which may cause imbalances between the nodes loads. Stability is a necessity for scalability.

2.6 Load-Balancing Approaches

Based on the different approaches used in implementing the four load distribution components, load-balancing techniques can be classified into the following two categories.

2.6.1 Static vs. Dynamic

Load-balancing policies can be broadly characterised as static, dynamic or adaptive. Static policies make task transfer decisions deterministically or probabilistically without taking in consideration the current state of the system. Static schemes assume previous knowledge of both application and system state (Braun, et. al,

2001). This approach can be effective when the workload can be sufficiently well characterised before the actual execution, but it fails to adjust to the fluctuations in system load. Static load balancing is advantageous in terms of low overhead as the decision is only made once before computation. However, it cannot adapt to changes of application requirement and system state.

Dynamic load-balancing is proposed to make decision based on the current system state and can rapidly adapt to workload fluctuations (Torrellas, et. al, 1995, Tzen and Ni, 1993). Dynamic load-balancing algorithm use system-state information to make load distribution decisions, so they have the potential to outperform static policies by improving the quality of their decisions. Essentially dynamic load distribution policies improve performance by exploiting short-term fluctuations in the system-state. Because they must collect state information of the system, they incur more communication overhead than their static counterparts, but this overhead is often well spent.

Adaptive load-balancing policies adapt their activities by dynamically changing their parameters, or even their algorithms, to suit the changing system state. While a dynamic policy takes system-state inputs into account when making its decisions, an adaptive policy takes system-state into account to modify either its parameters or the scheduling policy itself. The performance of load-balancing algorithms is very sensible to their parameters, which suggests that adaptive load distribution may be able to provide good performance when system-state changes widely (Becker and Waldmann, 1995, Casavant, 1988, Zhou, 1988, Shivaratri, et. al, 1992).

2.6.2 Centralised vs. Distributed

Load-balancing algorithms can be centralised, distributed or some hybrid form of both. Centralised means that information is collected at a single physical location at which all scheduling decisions are made. In centralised schemes, a central server collects load information and making decision based on global knowledge. Centralised schemes usually require global synchronisation to obtain global information at the cost of high synchronisation cost.

Distributed load balancing allows every machine to calculate its local view of load distribution and makes decision based on the partial knowledge. Typical distributed load balancing schemes are neighbour based, such as diffusion method (Subramanian and Scherson, 1994, Lovàsz and Winkler, 1995). The lack of global knowledge slows down the convergence rate of global balancing.

Even though some authors claim that centralised techniques achieve better results and are scalable with the size of the network, the great majority agrees that a load balancing policy must itself be distributed in order to avoid hotspots and thus be scalable. In this research work, Theimer and Lantz claim that the centralised approach is scalable and more efficient than a distributed one (Theimer and Lantz, 1989). However, they assume that most of the nodes are idle and the decentralised policy that has been studied here is based on each node broadcasting its load level to the system. They claim that if inter-processor communication is efficient and the system size is limited, the centralised approach to load information dissemination and task placement may be simple and effective (Theimer and Lantz, 1989). In

(Ozden, 1993), the author shows that a centralised scheme will scale only until a given number of nodes.

However, most authors argue that centralised strategies are not scalable because the central component is potentially a bottleneck, leading to less performance on larger networks (Kremien, and Kramer, 1992, Lüling and Monien, 1993). If one of the basic requirements of the load distribution strategy is to make no assumptions about the underlying network, including its size, as stated before, then it should be distributed in order to scale with system size. The functional capacity of any centralised server is bounded and cannot grow when the system where it is embedded is enlarged. To be independent of system size an algorithm should be completely distributed, taking advantage of distribution by maintaining only a partial view of the system at each node (Kremien, and Kramer, 1992, Lüling and Monien, 1993, Ozden, et. al, 1993).

The "Hicon" concept provides dynamic load balancing support on heterogeneous workstation networks (Becker and Zedelmayr, 1994, Becker, 1995). It employs distributed clustering, where each cluster is managed by a centralised load balancing component and inter-cluster load sharing is performed by using a decentralised policy. The centralised approach provides sophisticated load control, while the decentralised coupling of clusters ensures scalability.

Central load-balancing algorithms has no logical drawbacks, but it is not scalable as it will cause processing time overhead and delays to grow with system activity and

the number of processors. Above certain limit of system size, centralised approaches no longer offer efficient overall performance.

2.7 Load Distribution in Grid Networks

Grid Networks are typically a collection of various resources with different owners, but make it possible for users to develop complex applications that access remote nodes (Braun, et. al, 2001, Eager, et. al, 1986, Feng, et. al, 2000, Foster and Kesselman, 1999). Each of these nodes could be a uni-processor machine, a symmetric multiprocessor cluster, a distributed memory multiprocessor system, or a massively parallel supercomputer. Each node consists of a number of heterogeneous *resources*. The heterogeneity being in the type and capability of each of its resources (e.g., number of processors, CPU speed, amount of memory, and so on). One of the biggest advantages of a Grid Network is that it can offer resources to the users that are not locally available. With the Grid Network becoming a practical high performance computing alternative to the traditional supercomputing environment, several aspects of effective Grid Network resource utilisation are gaining significance.

Since Grid Networks may consist of millions of interconnected nodes, it is likely that some nodes would be heavily loaded while others are lightly loaded or even idle. To achieve maximum efficiency of these large systems, it is desired that the workload is distributed among all the nodes so that resource utilization is increased and maximum performance is achieved. A load distribution scheme must decide where and when a given task should be executed in order to increase performance. Therefore, implementing an effective load-balancing paradigm that would be

integrated with Grid Networks for efficient resource discovery and load distribution will have an important role in designing self-configuring and self-optimising Grid Networks. With its huge number of resources, an efficient load-balancing scheme can lead to improve overall system performance.

Due to its essential role in high performance computing, the field of load-balancing has been active for decades and many techniques and problem formulations have been used for parallel and distributed systems (Mitzenmacher, 2001, Lüling, et. al, 1991, Peixoto, 1996, Litzkow, et. al, 1988, Montresor, et. al, 2001). However, the majority of these works focus on traditional distributed systems and are no longer suitable for the Grid Networks (Kremien and Kramer, 1992, Lüling and Monien, 1993). Some techniques use a centralised approach that leave the algorithm un-scalable, while others assume the overhead involved in searching for appropriate resources to be negligible. Furthermore, classical load-balancing algorithms do not consider a Grid Network node to be *k*-resource nodes and only work towards maximising the utilization of one of the resources

The Grid Network system is different from the traditional distributed systems as it converts high performance computing platforms into heterogeneous, dynamic and shared platforms, which prevents conventional load-balancing schemes from benefiting large-scale parallel applications. Therefore, extending existing load-balancing schemes to handle one or more of the challenges of Grid Networks (such as heterogeneity, resource sharing, high latency and dynamic system state) is a challenging research area.

Despite the fact that several load balancing techniques are proposed in the literature, much of the analysis of these techniques is limited to the results obtained from simulation; few have been studied using analytical methods and even fewer from measuring the performance of an actual Grid Network. Moreover, there is a need to consider the cases where services are time-bounded, or require certain QoS requirements, or when jobs are depending on the output of other jobs.

2.8 Summary

A relevant literature review in the area of Grid Networks technology is presented in this chapter. Research challenges and performance optimization techniques required to achieve the self-optimization and self-configuration properties of Grid Networks are also discussed. Then, the load-balancing problem is described and a brief review of the most common load-balancing solutions and strategies is summarised. Additionally, the load-balancing stability and scalability issues have also been discussed.

Although intensive work has been done in the area of load balancing, the Grid Network environment is different from the traditional network systems, which prevents existing load-balancing schemes from benefiting large-scale networks. The Grid Network evolves high performance computing platforms to heterogeneous, dynamic and shared environments, which prevent exploiting conventional load balancing techniques directly. Thus how to adapt current load-balancing schemes to Grid Network becomes the focus of load distribution research area.

Chapter 3

A REVIEW OF COMPLEX NETWORKS THEORY

Chapter 3

A REVIEW OF COMPLEX NETWORKS THEORY

3.1 Introduction

Complex Networks Theory is the field that describes the structural and dynamic properties of networks. A complex network is a network that has certain significant topological features which do not occur in simple networks. Such topological features include: a heavy tail in the degree distribution; a high clustering coefficient; and an indication of a hierarchical structure. Simple networks do not have such properties, and are typically described by graphs such as random graphs, which show a high similarity regardless of what part is examined. In mathematics, a random graph is a graph that is generated by some random process. A graph is a thcoretical representation of a set of objects where some pairs of the objects are connected by links. The interconnected objects are represented by mathematical abstractions called nodes, and the links that connect some pairs of nodes are called edges. Many existing networks (such as social, computer, and neural networks) can be classified as complex networks based on their topological structure.

In the past few years, there has been a great achievement in the field of complex networks due to combining ideas and analytical tools from statistical mechanics. These analytical tools facilitated the development of a number of protocols and

models for complex networks that result in predictable properties. Networks have a variety of dynamical features (such as self-organising, robustness, heterogeneity, connectedness of nodes, etc.) and their topology structure determines these dynamical features.

Several theories and models have been developed to analyse complex networks, and to characterise the dynamic properties of networks. For example, the *Random Graph Theory* described by Erdös and Rényi in1930s (Erdös and Rényi, 1959), and the analysis of social networks started in the 1960s and 1970s (Scott, 2001). Graph theory has long since been described as part of system theory (Laue, 1970). Several authors have discussed the characteristics of nature and society networks (Bollobás, 1985; Barabási, 2002; Bornholdt & Schuster, 2003; Buchanan, 2002; Dorogovtsev & Mendes, 2003; Huberman, 2001; Watts, 1999). As network science has continued to grow in importance and popularity, other models of complex networks have been developed.

Traditionally, complex networks have been described using Graph Theory. Simple networks with a large degree of design can be described by regular graphs, but large-scale networks are often too complex for their designing principles to be visible. The first approach is to model them as completely random, and to study the properties of random graphs with the same number of nodes and edges as the original network. Then, the properties of the corresponding random graphs may give an idea about the properties of the real networks.

Random Graph Theory was the simplest theory to describe complex networks. Pál Erdős and Alfréd Rényi were the first who studied random graphs (Erdös and Rényi, 1959, Erdös and Rényi, 1960). According to the Erdős-Rényi (ER) model, we start with N nodes and connect every pair of nodes with probability p. At the end of this process, the graph will have approximately $pN(N - 1)/2$ edges distributed randomly. This model has been used in various fields dealt with complex networks.

Another two important models for complex networks are the scale-free networks model (Barabási and Albert, 1999), and small-world networks model (Watts and Strogatz, 1998).

In scale-free networks model, some nodes have higher degree, and most of the nodes are of lower degree. Scale-free networks' structure and dynamics are independent of the number of nodes the network has. In other words, a network that is scale-free will have the same properties no matter how many nodes it has. The degree distribution for scale-free networks follows a power law relationship (Barabási and Albert, 1999).

In small-world networks model, most of the nodes are not neighbours, but most of the nodes can be reached from every other by a small number of hops or steps. The small-world networks model is classified based on two independent structural features; the clustering coefficient and average shortest path length (Watts and Strogatz, 1998). ER random graphs have a small average shortest path length along with a small clustering coefficient. However, the small-world model has a small average shortest path length, and a large clustering coefficient.

Many real world networks are modelled as small-world networks. Social networks, the connectivity of the Internet, and gene networks all show small-world network characteristics (Watts and Strogatz, 1998).

The Internet and the World Wide Web inspired the development of the field of network analysis, or complex networks, in statistical physics. Without the availability of digitised data, none of the examples used in complex networks theory could have been exploited. Availability of data in digitised form facilitates computation in particular for huge networks. The development of this speciality started with the empirical investigations of different types of real-world networks, such as the Internet and the web. It was extended into model-building activities that tried to mirror the statistical features found empirically.

3.2 Complex Networks Concepts

There have been dramatic advances in analysing complex networks in the literature, and many quantities and measures have been proposed and investigated in depth. However, four main concepts emerged in the past few years that occupy a major part in investigating complex networks. These four concepts are small-world concept, degree distribution, clustering coefficient, and diameter.

3.2.1 The Small-World Concept

The small-world network is a class of random graphs, where every node can be reached from other nodes by a small number of hops or steps. The small-world concept refers to the fact that in most networks, despite the large number of nodes they are made of, the typical distance between any two nodes is very short (Milgram,

1967). Here the distance between any two nodes in a network is the number of links in the shortest path connecting them. Therefore, distances do not depend on the distances between the nodes; it only depends on the interconnections between them.

Small-world networks were identified as a class of random graphs by Duncan Watts and Steven Strogatz in 1998 (Watts and Strogatz, 1998). They noted that graphs could be classified according to their clustering coefficient and their mean-shortest path length. Small-world networks, as compared to other random graphs with the same number of nodes and edges, are characterised by clustering coefficients significantly higher than expected, and mean shortest-path length lower than expected.

Moreover, Erdős and Rényi have shown that if the connection probability p is larger than $\ln(N)/N$, almost any random graph will be strongly connected. Hence, we can find a path of edges connecting any two nodes in the system. The typical distance between any two nodes in a connected random graph tends to be small, and it scales as the logarithm of the number of nodes. Therefore, such random graphs are considered as small-worlds networks.

3.2.2 Degree distribution

The degree distribution gives the probability distribution of nodes' degrees (edges) in a network. The degree of a node (or connectivity), k, gives an indication of how many links (or edges) the node has to other nodes. Its use comes from the study of random graphs by Paul Erdős and Alfréd Rényi (Erdös and Rényi, 1959), and it has become an important concept, which describes the topology of complex networks.

In directed networks, nodes have two types of degrees. The incoming degree k_{in} denotes the number of links that point to a node, and an outgoing degree k_{out} denotes the number of links that start from it. An undirected network with N nodes and L links is characterised by an average degree $\bar{k} = 2L/N$ (Barabási and Oltvai, 2004). Otherwise, both incoming and outgoing links are counted in the undirected case (Freeman, 1979).

The degree distribution of a graph is a function describing the total number of nodes in a graph with a given degree (or number of edges). Not all nodes in a network have the same number of edges. Thus, there is always a spread in the number of edges associated with a given node. This spread is characterised by a distribution function $P(k)$, which gives the probability that a randomly selected node has k edges.

Since in a random graph every node is equivalent, the majority of the nodes have approximately the same number of edges, equal with the average degree $\bar{k}$ of the network. Consequently, the degree distribution is a Poisson distribution with a large peak at $P(\bar{k})$ (Albert and Barabási, 2002).

An observation that recently has attracted much attention was that for most large networks the degree distribution does not follow the Poisson distribution expected for random graphs. In particular, for a large number of networks, including the World Wide Web (Albert et al., 1999), Internet (Faloutsos et al., 1999), or cellular networks (Jeong el al., 2000), the degree distribution has a power-law tail, which expressed as

$$P(k) \sim k^{-\lambda} \tag{3.1}$$

where λ is the exponent of the power-law (Barabási, 2002) . A power-law degree distribution indicates that a few nodes hold together numerous small nodes. Thus, the networks that are characterised by a power-law degree distribution are highly non-uniform; most of the nodes have only a few links. A few nodes with a very large number of links hold these nodes together.

Other networks, such as an electrical power grid (Amaral et al., 2000), display an exponential tail. Therefore, the degree distribution is a common way of classifying graphs into categories, such as random graphs (Poisson distribution) and scale-free networks (Power law distribution) (Albert and Barabási, 2002).

3.2.3 Clustering coefficient

Duncan Watts and Steven Strogatz (Watts and Strogatz, 1998) introduced the term clustering coefficient to determine whether a graph is a small-world network or not.

Let i be a node in the network with k_i edges that connect it to k_i other nodes. So, if the first neighbours of the node i were also a part of this network, then, there would be $k_i(k_i - 1)/2$ edges between them. Therefore, the clustering coefficient (C_i) of node i is the proportion of edges between the nodes within its neighbourhood divided by the number of edges that could possibly exist between them. Accordingly, the ratio between the number of edges that actually exist between these k_i nodes, E_i, and the total number $k_i(k_i - 1)/2$ gives the value of the clustering coefficient of node i. Thus,

$$C_i = \frac{2E_i}{k_i(k_i - 1)} \tag{3.2}$$

So, the clustering coefficient of the whole network is then the average of all individual clustering coefficients,

$$C = \frac{1}{N}\sum_{i=1}^{N} C_i = \frac{1}{N}\sum_{i=1}^{N} \frac{2E_i}{k_i(k_i - 1)} \tag{3.3}$$

The clustering coefficient of random graphs is small. In fact, since the edges are distributed randomly, the probability that the first neighbours of a node are connected is the same that any two nodes in the graph are connected. Consequently, the clustering coefficient of random graphs is $C = p$. However, Duncan Watts and Steven Strogatz mentioned that in most real networks, the clustering coefficient is much larger than it is in a random network with the same number of nodes and edges (Watts and Strogatz, 1998). This observation was the first indication that real networks have properties that go beyond random graphs.

3.2.4 Connectedness and diameter

The diameter of a graph is the maximum distance between any pair of its nodes. The connectivity and diameter of a random graph has been studied by many authors (Chung and Lu, 2001, Newman, 2001, Watts and Strogatz, 1998). They observed that for most values of connection probability p, almost all graphs with the same N and p have identical diameters. Thus, for all graphs with N nodes and connection probability p, the range of values in which the diameters of these graphs can vary is very small, and usually concentrated around $\ln(N)/\ln(\bar{k})$.

In most cases, random graphs are likely to have small diameters, provided that probability p is not too small. This is because the diameter of a random graph depends logarithmically on the number of nodes the graph. Therefore, if the connection probability is not large enough, random graph tends to be spreading.

The connectedness of a random graph can be summarised using the following important results:

- If $\bar{k} = pN < 1$, then a typical graph is composed of isolated trees and its diameter equals the diameter of a tree.
- If $\bar{k} = pN > 1$, then a giant cluster appears. The diameter of the graph equals the diameter of the giant component.
- If $\bar{k} \geq 3.5$, then the diameter of the graph is proportional to $\ln(N)/\ln(\bar{k})$.
- If $\bar{k} = pN \geq \ln(N)$, then almost every graph is totally connected. The diameters of the graphs are concentrated on a few values around $\ln(N)/\ln(\bar{k})$.

Another way to describe the connectivity of edges of random graph is to calculate the average distance between any pair of nodes, or the average path length. It is expected that the average path length scales with the number of nodes in the same way as the diameter. Therefore, the average path length of a random graph can be determined by Equation (3.4) (Albert and Barabási, 2002).

$$l_{rand} \square \frac{\ln(N)}{\ln(\bar{k})} \tag{3.4}$$

In most cases, random graphs are likely to have small diameters, provided that probability p is not too small. This is because the diameter of a random graph depends logarithmically on the number of nodes the graph. Therefore, if the connection probability is not large enough, random graph tends to be spreading.

The connectedness of a random graph can be summarised using the following important results:

- If $\bar{k} = pN < 1$, then a typical graph is composed of isolated trees and its diameter equals the diameter of a tree.
- If $\bar{k} = pN > 1$, then a giant cluster appears. The diameter of the graph equals the diameter of the giant component.
- If $\bar{k} \geq 3.5$, then the diameter of the graph is proportional to $\ln(N)/\ln(\bar{k})$.
- If $\bar{k} = pN \geq \ln(N)$, then almost every graph is totally connected. The diameters of the graphs are concentrated on a few values around $\ln(N)/\ln(\bar{k})$.

Another way to describe the connectivity of edges of random graph is to calculate the average distance between any pair of nodes, or the average path length. It is expected that the average path length scales with the number of nodes in the same way as the diameter. Therefore, the average path length of a random graph can be determined by Equation (3.4) (Albert and Barabási, 2002).

$$l_{rand} \square \frac{\ln(N)}{\ln(\bar{k})} \tag{3.4}$$

Albert and Barabási showed that the average path length of real networks is close to the average path length of random graphs with the same size (Albert and Barabási, 2002). By comparing the predicted average path length for random graphs of different sizes with data collected from several real networks, they found that the trend of the data in real networks is similar with theoretical prediction of random graphs, and that for several networks Equation (3.4) works quite well. But in general the average path length of real networks is larger than that of a random graph with the same N and $\bar{k}$.

3.3 Summary

The specialty of complex networks theory remains an interesting development that scientists should be aware of. It concerns new definitions for connectivity and new indicators for network analysis. It also concerns results about the functionality of connectivity, which has implications for the accessibility of information in networks and the functional stability of this information. Possible explanations of connectivity with the help of mathematical models require further qualitative and context bounded research into the nature of complex networks.

In this chapter, the various theoretical tools developed to model the complex networks have been reviewed. The four robust measures of the network topology; small-world, clustering coefficient, degree distribution and diameter were the basis of network modelling in the past few years, resulting in the introduction and study of three main classes of modelling paradigms. A review of network modelling efforts and the theoretical developments of the various models and theoretical tools are discussed in the following chapter.

Chapter 4

COMPLEX NETWORKS MODELS

Chapter 4

COMPLEX NETWORKS MODELS

Many researchers have proposed models of networks that try to explain either how networks come to have the observed structure, or what the expected effects of that structure will be. A summary of these models is presented, covering random graph models and their generalisations, the small-world model, and models of network growth, particularly the preferential attachment and scale-free models.

4.1 Random Graph Networks Model

Classical models of networks share the assumption that the connections between the nodes occur at random. The first attempt at developing a model for large and random networks was the *Random Net* (Rapoport, 1957, Solomonoff and Rapoport, 1951). Random Net was concerned with social networks of relationship among groups of people (such as the patterns of friendships between individuals, business relationships between companies, and intermarriages between families). Later, this work was independently rediscovered by Erdös and Rényi and gave it the name *Random Graph* (Erdös and Rényi, 1959, Erdös and Rényi, 1960).

In Eröds and Rényi (ER) random graph model, the probability of having a graph G with N nodes and k edges follows a Binomial distribution, and is given by

$$P_{N,k,p}(G) = p^k (1-p)^{\left(\frac{N(N-1)}{2}-k\right)} \tag{4.1}$$

Here, every possible edge occurs independently with connection probability p. At the end of this process, the graph will have approximately $pN(N - 1)/2$ edges distributed randomly.

In a large random graph, there are several nodes with the same degree, and the number of nodes with a given degree (or the degree distribution) can be calculated. So, in a random graph with connection probability p, the number of nodes with degree k is

$$P(k) = C_{N-1}^{k} p^k (1-p)^{N-1-k} \tag{4.2}$$

where C_{N-1}^{k} is the probability space in which k edges are chosen from the total number of edges $N-1$.

Thus, for large N, the number of nodes with degree k follows a Poisson distribution (Erdös and Rényi, 1960, Bollabás, 2001),

$$P(k) \square e^{-pN} \frac{(pN)^k}{k!} = e^{-\bar{k}} \frac{(\bar{k})^k}{k!} \tag{4.3}$$

Figure 4.1 shows the degree distribution for a random graph with $N = 10,000$ nodes and connection probability $p = 0.0015$. The plot compares the calculated number

of nodes with degree k, with the expectation value of the Poisson distribution $P(k)$.

As shown in the figure, the difference between them is small.

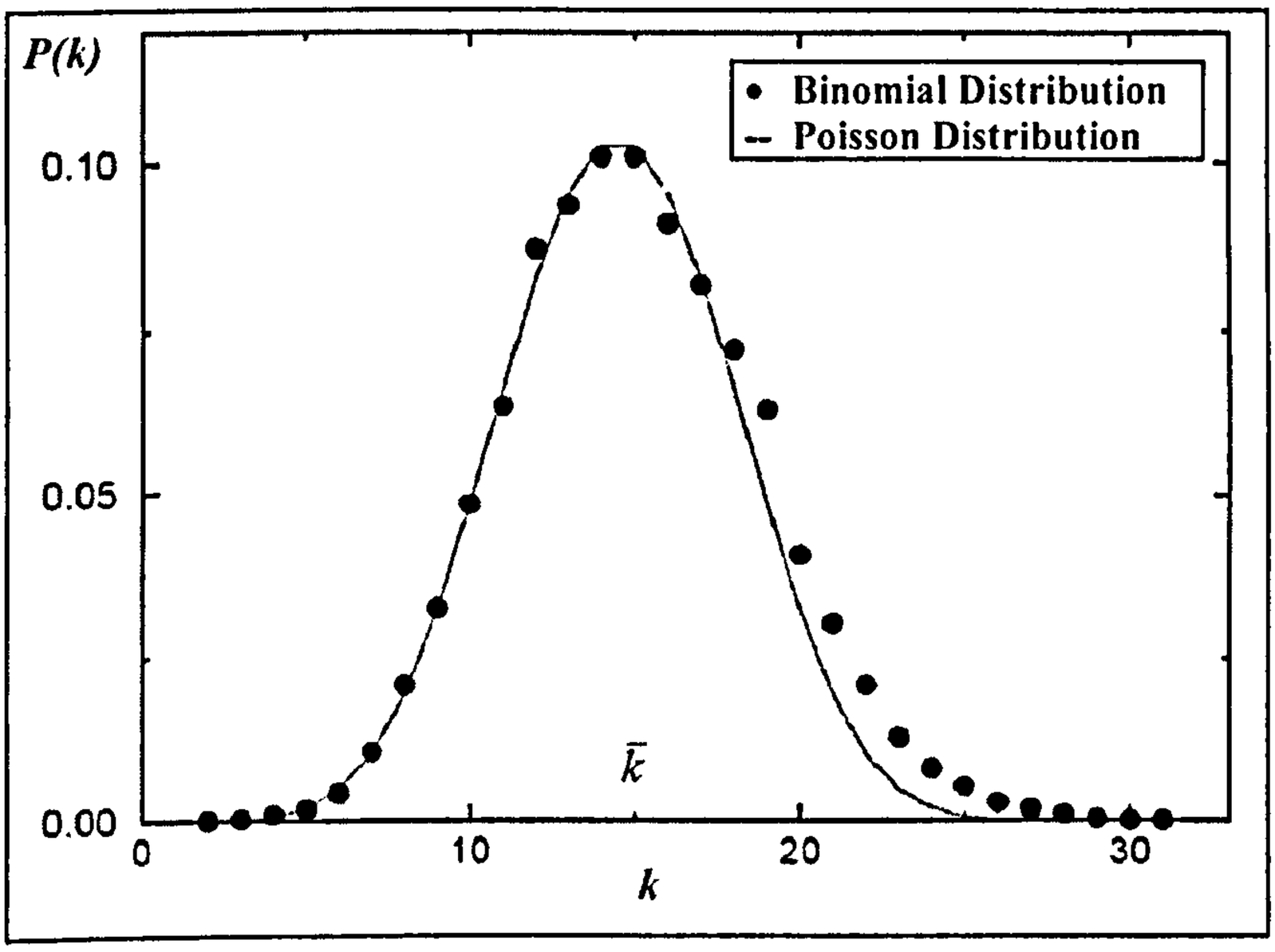


Figure 4.1 The degree distribution for a random graph with $N = 10,000$ nodes and connection probability $p = 0.0015$. After (Albert & Barabási, 2002).

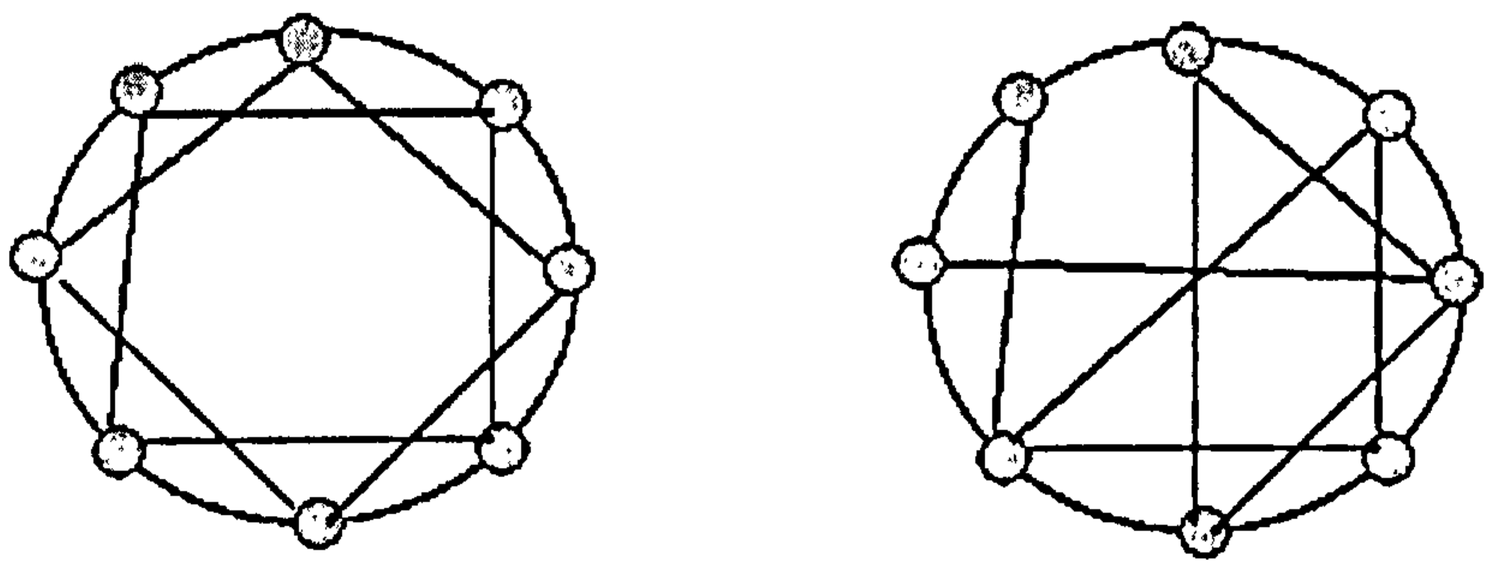
Thus, in ER model, the probability that an ER graph has more or less than the expected number of edges decreases exponentially. This binomial distribution implies that each node will have a degree, which is close to the average degree, and that the number of nodes with much higher or much lower degree than average is very small. Thus, the probability that any node has the expected number of edges is the same.

The ER random graph has been used in various fields dealing with complex networks. Though, do real world networks (such as the Internet and Web) have a completely random structure without any organising characteristics? It is easy to argue against a fully random structure for many network systems, as they must

display some organising properties that allow them to operate successfully. Therefore, if these networks differ from a random topology, new tools and measures are needed to realise the underlying organising properties.

4.2 Small-World (SW) Networks Model

The small-world networks concept was introduced by Watts and Strogatz, to describe networks where nodes are linked to each other by only a few nodes in between (Watts and Strogatz, 1998, Watts, 1999). Small-world networks can be constructed from regular networks just by adding random elements to them (Buchanan, 2002); see Figure 4.2.



(a) Regular Network **(b) Small-World Network**

Figure 4.2 Schematic representation of the creation of Small-world networks from Regular networks. (a) Regular networks. (b) Small-world networks created after some steps of random rewiring of links. After (Buchanan, 2002).

To create a small-world network, we start with a ring network with N nodes in which every node is connected to its first $\bar{k}$ neighbours. Then, we randomly rewire each edge of the network with probability p such that self-connections and duplicate edges are excluded. By varying p, the transition between regular ($p=0$) and randomness ($p=1$) in networks can be observed; see Figure 4.3. Hence, the

small-world networks are located somewhere in the middle between regular graphs and random graphs.

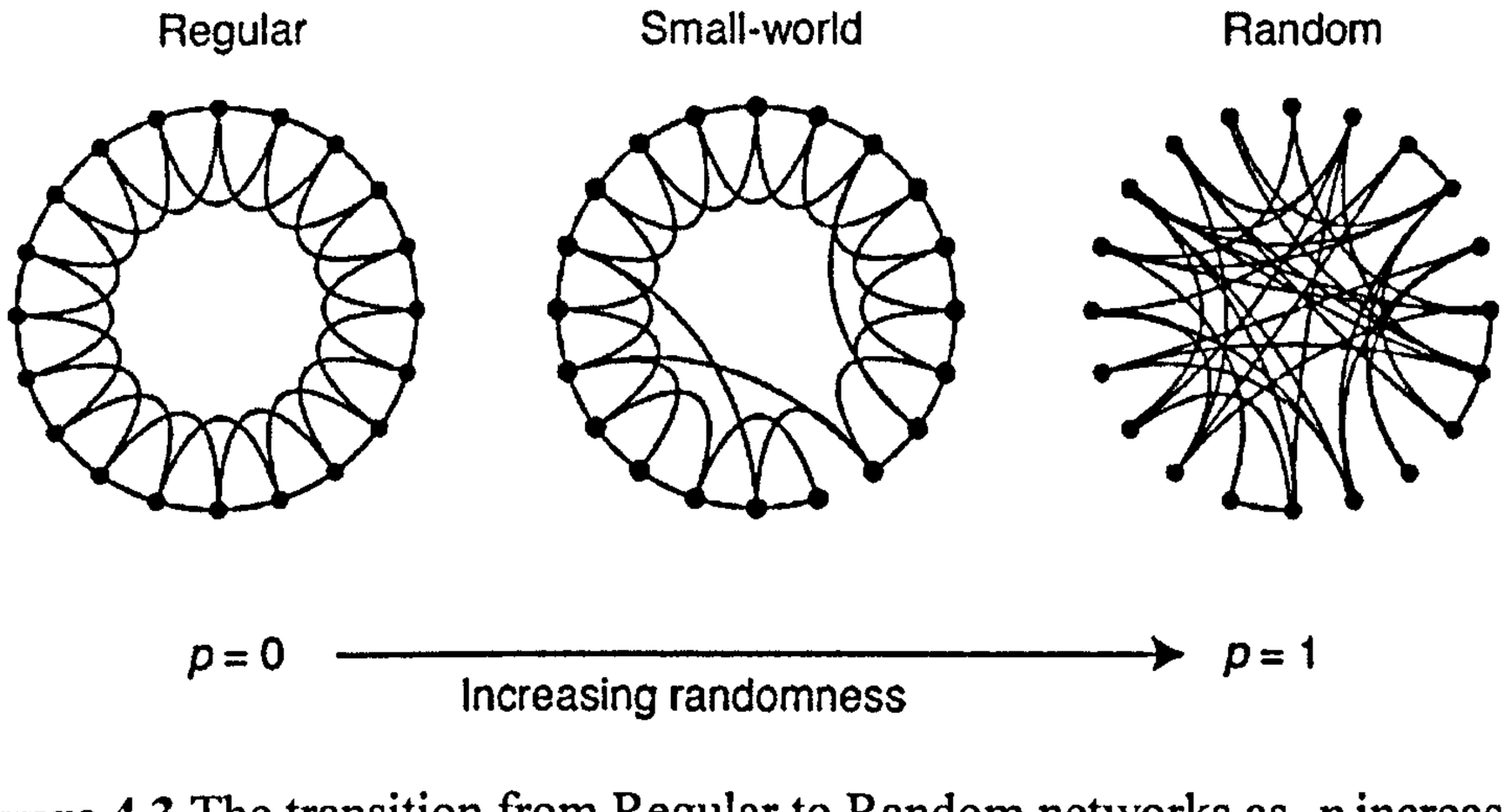


Figure 4.3 The transition from Regular to Random networks as p increases. After (Watts and Strogatz, 1998).

Alternatively, small-world networks can be defined by examining the local neighbourhood and the diameter of the network. The diameter of a network is the average shortest path length of a network. Small-world networks are then defined as having local neighbourhood similar to regular networks, and the diameter of the network increases logarithmically with the number of nodes as in random networks (Amaral et al., 2000; Hayes, 2000).

In random networks, the distance between any two nodes can be small. Here, this distance is number of links that connect the nodes along the shortest path. Small-world networks share these properties with random networks. The difference between them is that small networks have a higher clustering coefficient, C. The clustering coefficient of a network describes at which degree the nodes that are connected to a certain node are also connected to each other. It compares the number

of existing links in a neighbourhood of a node with the number of all possible links in that neighbourhood. The clustering coefficient of real world networks is typically compared with the clustering coefficient for a random network (Strogatz, 2001).

Figure 4.4 shows the degree distribution of the small-world model for $\overline{k} = 6$ and various p. And for a comparison, the degree distribution of a random graph with the same parameters is plotted with filled symbols. The symbols are obtained from numerical simulations of the small-world model with $N = 1000$ (Barrat & Weigt, 2000). We can see that the degree distribution for small-world network is similar to that of a random graph, and only when $k \geq (\overline{k}/2)$ values are present. Moreover, it has a peak at $\overline{k}$, and it decreases exponentially for large k. Thus, in small-world model, when $p = 0$, each node has the same degree k, and the degree distribution is a delta function centred at k. A non-zero p introduces disorder in the network, broadening the degree distribution while maintaining the mean degree equal to $\overline{k}$. As a result, all nodes will have approximately the same number of edges.

4.3 Scale-Free (SF) Networks Model

There are two features of real networks that are not included in the ER and SW models. First, both models assume that we start with a fixed number of nodes (N) which are then randomly connected (ER model), or reconnected (SW model), without modifying N. However, most real world networks are created by the continuous addition of new nodes to the system and the number of nodes can change over time. Thus, many networks start with a small number of nodes, then the number of nodes increases throughout the lifetime of the network.

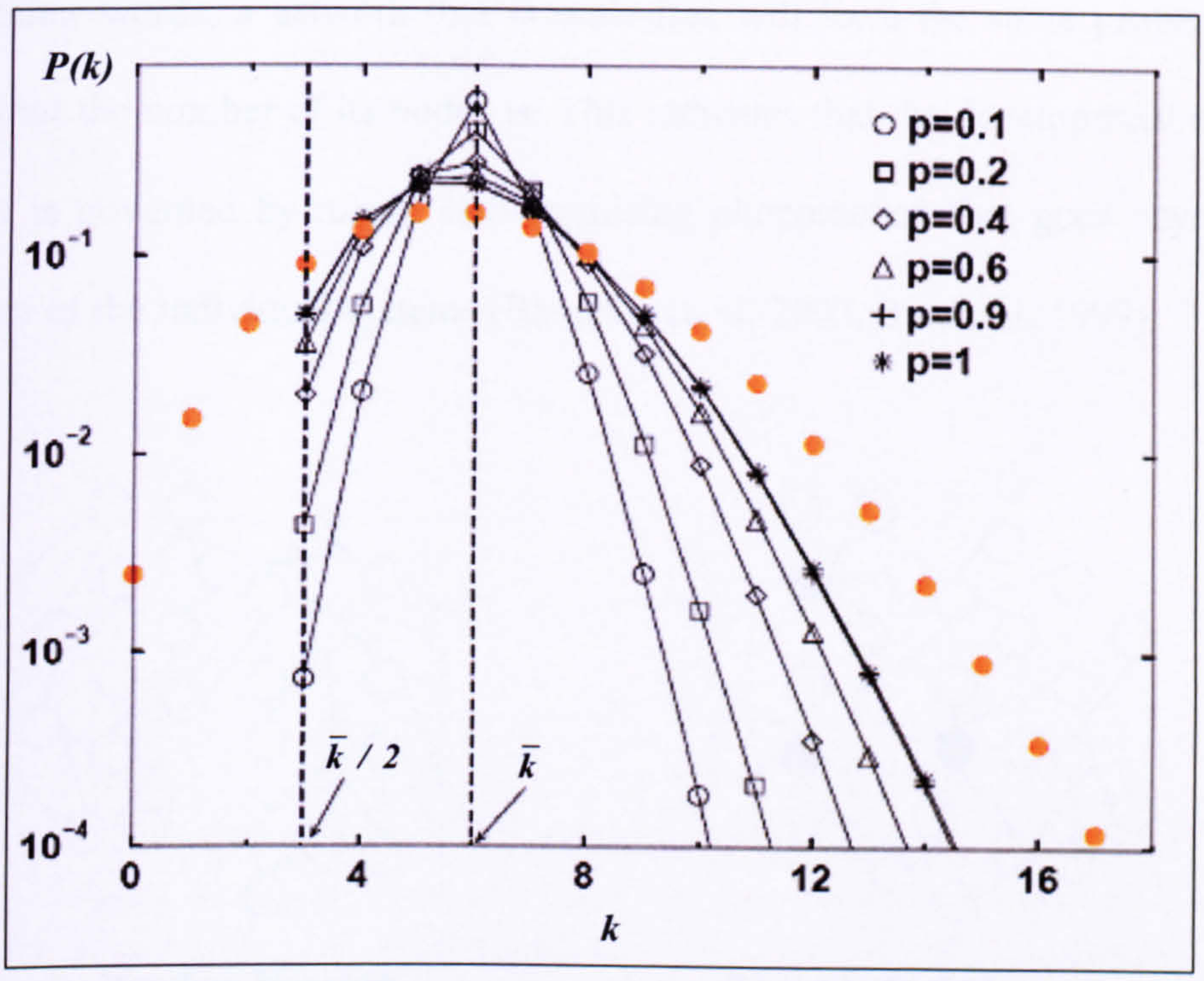
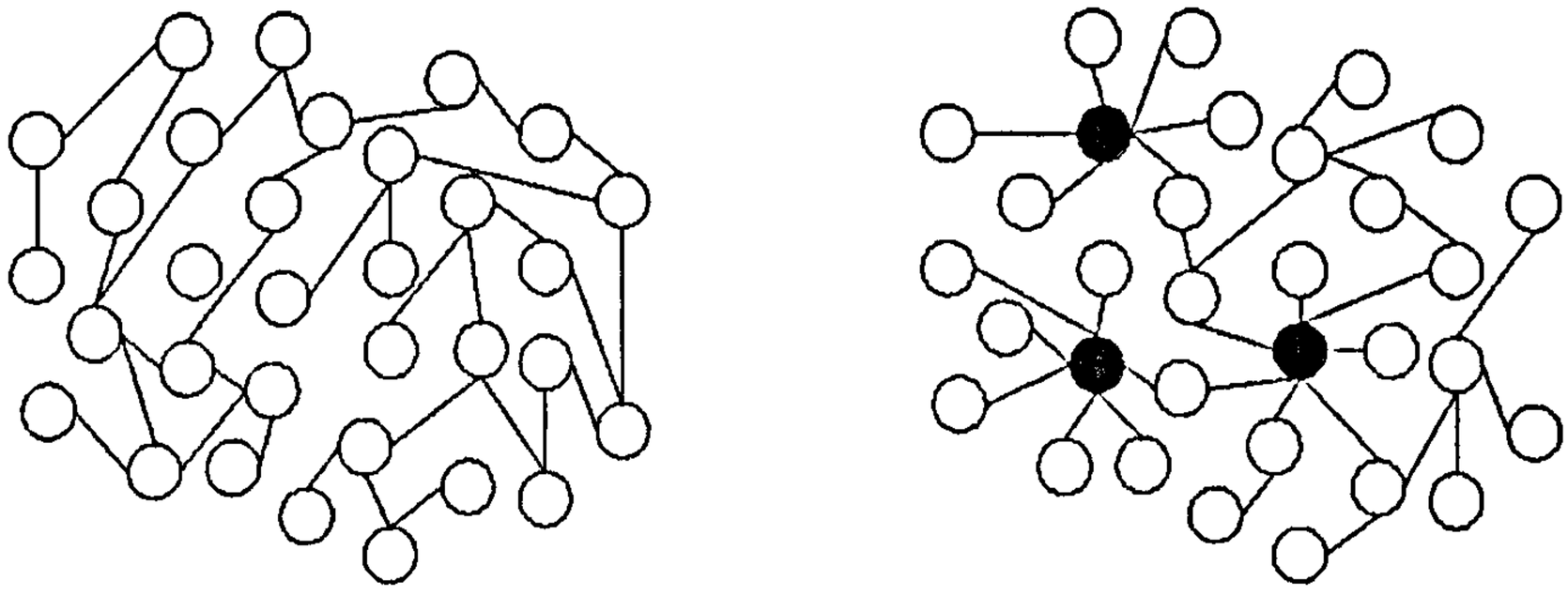


Figure 4.4 The Degree distribution of the small-world model for $\bar{k} = 6$ and various values of connection probability p. After (Barrat & Weigt, 2000).

Secondly, the previous random network models assume that the probability that two nodes are connected is random and uniform. However, many real networks show preferential connectivity. For example, a newly created webpage will likely include links to well known, popular documents which already have high connectivity. Therefore, the probability that a new node connects to the existing nodes is inconsistent, but there is a higher probability to be linked to a node that already has a large number of connections.

A model based on these two features leads to the observed scale-free distribution since networks expand continuously by the addition of new nodes, and new nodes attach preferentially to already well-connected nodes; see Figure 4.5. The scale-free network structure and dynamics are independent of the number of nodes the network

has. In other words, a network that is scale-free will have the same properties no matter what the number of its nodes is. This indicates that the development of large networks is governed by robust self-organising phenomenon that goes beyond the particulars of the individual systems (Barabási et. al, 2003, Barabási, 1999).



(a) Random Network **(b) Scale-Free Network**

Figure 4.5 Examples of a Random network and a Scale-free network. Each network has 32 nodes and 32 links. (a) Random network. (b) Scale-free network. After (Castillo, 2004).

Therefore, and based on the two common features; *growth* and *preferential attachment,* a new model has been developed to characterise the *scale-free Power-law* degree distribution observed in real networks. This model is called a Scale-Free (SF) model (Barabasi and Albert, 1999).

Albert and Barabási introduced the concept of a scale-free network to describe a specific distribution of links over nodes (Albert & Barabási, 2002). They demonstrated that many large networks share the common feature that their degree distribution follows a power-law for large k, and that random graph theory and the small-world model cannot reproduce this feature. Moreover, even for networks with

degree distribution that has an exponential tail, their degree distribution significantly differs from Poisson degree distribution; see Figure 4.6.

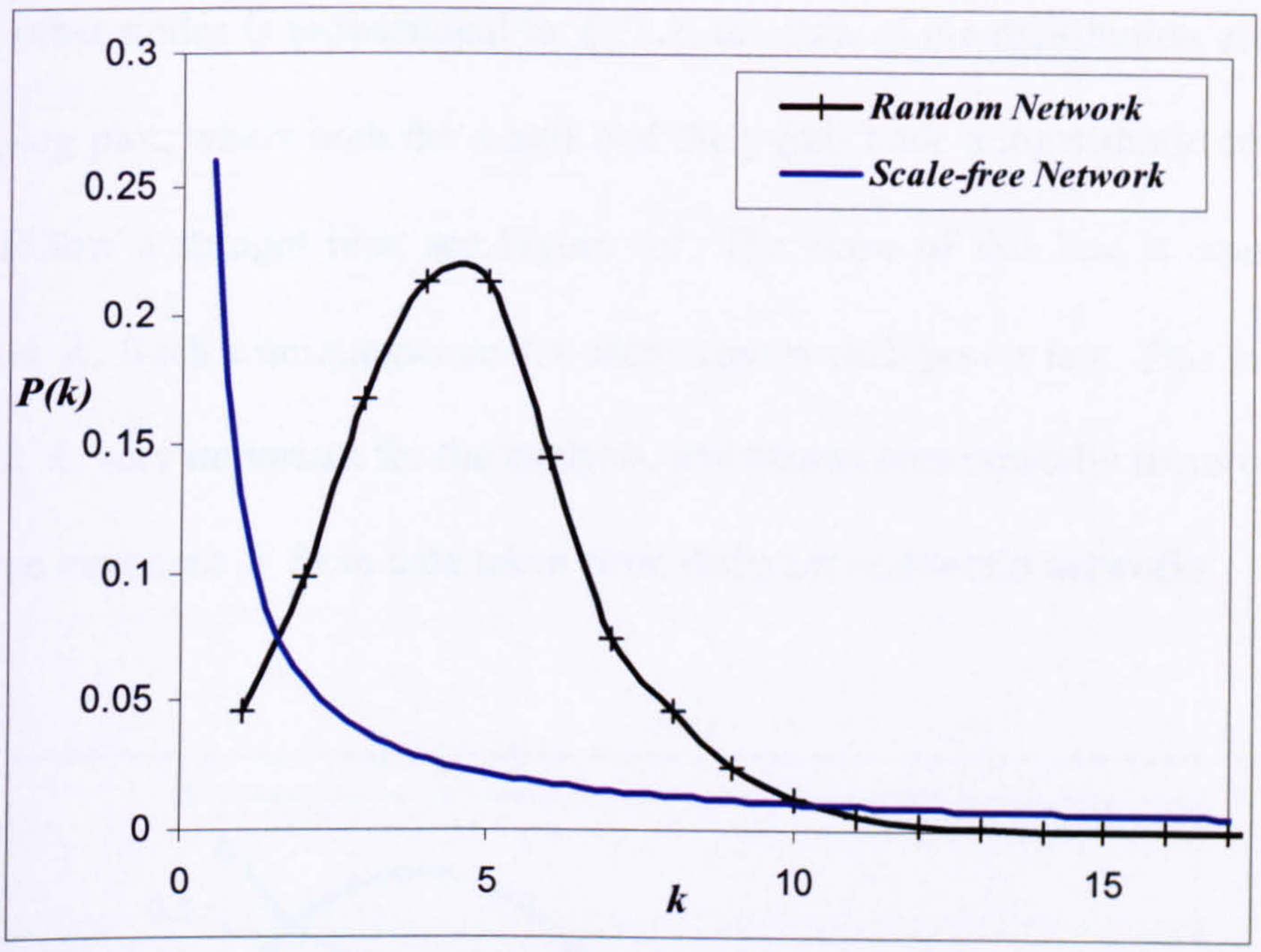


Figure 4.6 Comparison of Degree distribution for Random networks and Scale-free networks. After (Scharnhorst, 2003).

The scale-free model displays an extremely skewed distribution with a long tail that can be described by a Power-law. The power-law distribution implies that most of the nodes have only few links, and few nodes have a very large number of links (Barabási, 2001). Power laws characterise the transition from disorder to order. Therefore, the discovery of power law is particularly important because it comes together with the presence of self-organising mechanisms. Barabási mentioned that "through these findings, complex networks were lifted out of the jungle of randomness where Erdös and Rényi had placed them forty years earlier and dropped them into the centre of a colourful and conceptual rich arena of self-organisation" (Barabási, 2002).

There is no characteristic scale for node's degree in networks with skew distribution (Krapivsky and Redner, 2002). For this reason, skew distributions are called scale-free. Power-law distribution implies that the probability of finding a node with k links to other nodes is proportional to $k^{-\lambda}$. If the data of the distribution are plotted in a log-log plot, where both the x-axis and the y-axis have a logarithmic scale, they should follow a straight line; see Figure 4.7. The slope of this line is equal to the parameter λ. Such a unique parameter characterises each power law. This makes the exponent λ very important for the analysis, and researchers typically focus on fitting the degree exponent λ from data taken from different real world networks.

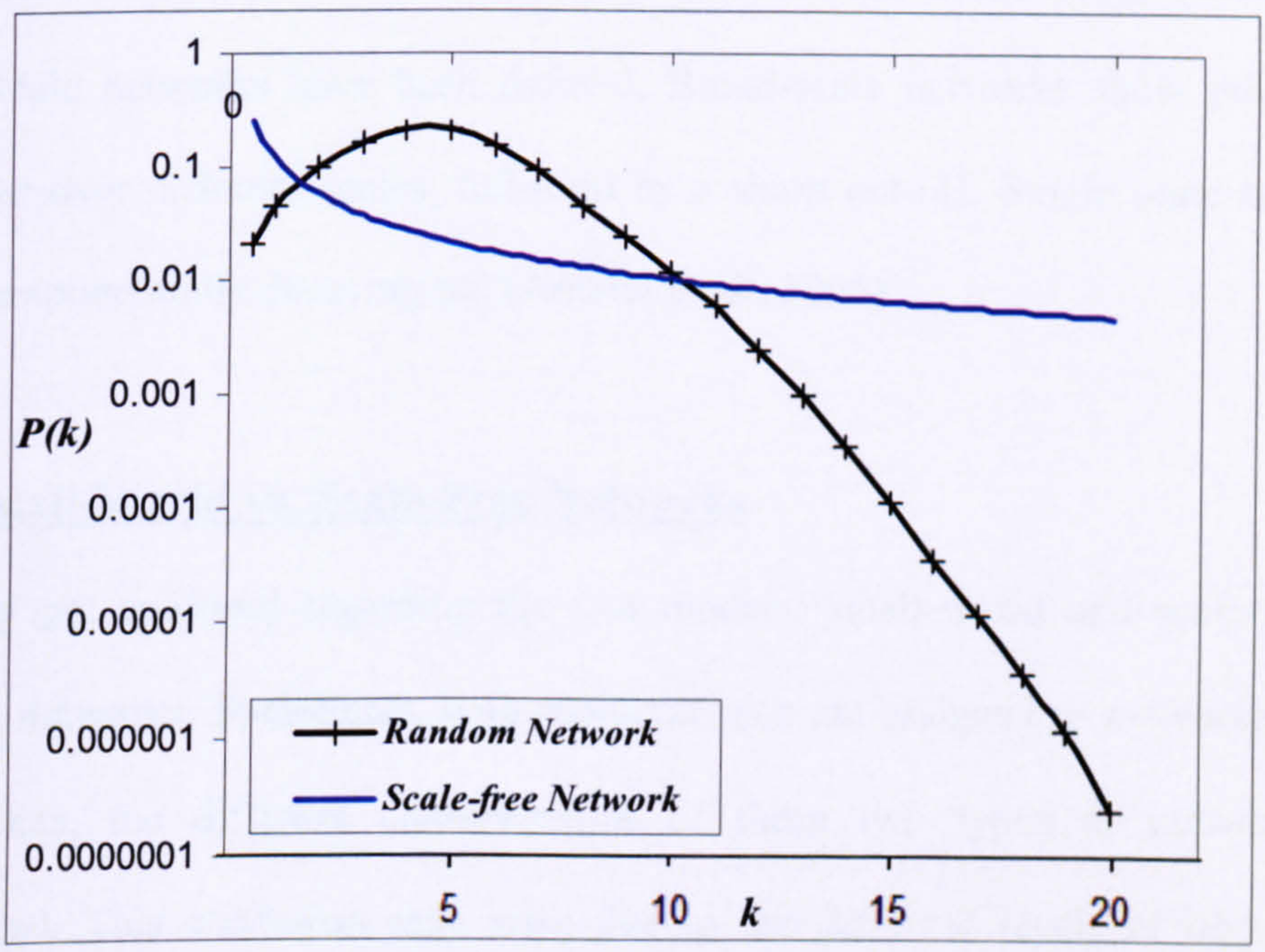


Figure 4.7 The logarithmic scale of the degree distribution for Random networks and Scale-free networks. After (Scharnhorst, 2003).

The different values of the exponent indicate the different dynamic mechanisms working behind the distributions; similar values indicate the action of similar

mechanisms. Furthermore, the resistance of networks against attacks (i.e. the removal of nodes or links) appears to depend on the value of the degree exponent.

As the power-law observed for real networks describes systems of rather different sizes at different stages of their development, it is expected that a correct model should provide a distribution whose main features are independent of time. The development of the power-law scaling in the model indicates that growth and preferential attachment play an important role in network development.

However, the power law cannot always be clearly observed in real data. Therefore, other types of distributions have been introduced, such as Broad-Scale networks and Single-Scale networks have been defined. Broad-scale networks show power law behaviour over different scales, followed by a sharp cut-off. Single scale networks have an exponentially decaying tail (Amaral et al., 2000).

4.4 Small-World vs. Scale-Free Networks

One may get confused regarding the two models: small-world and scale-free for complex networks. Sometimes, both characteristics are assigned to networks and in other times, the different characteristics of these two types of networks are highlighted. This confusion may arise due to the different levels of observation which are being applied. Mathematical models, as theoretical justification for experimental facts, follow one type of reasoning. Experimental analysis of real world networks follows another type of reasoning. Therefore, to find proper descriptions for both small-world networks as scale-free networks, prototypes of models have been developed at the theoretical level. Small-world networks are represented by the

Watts-Strogatz (WS) model and scale-free networks by the Barabási-Albert (BA) model.

Albert and Barabási mentioned that the WS model has a degree distribution similar to a random graph and is not scale-free (Albert and Barabási, 2002). Holme mentioned that the WS model has high clustering, whereas the BA model has a clustering coefficient that scales toward zero for large number of nodes (Holme et al., 2002). The BA model produces a scale-free network, which is not a small-world network. However, it shares one feature with a small-world network, which is that two randomly chosen nodes are connected by a very short path. Furthermore, to make the situation even more complex, other models have been developed which show both properties.

In terms of experimental analysis, real networks usually display various degree distributions and sometimes do not meet the exact criteria of a Power-law. Moreover, when the network has small number of nodes and links, it is more difficult to statistically analyse its properties. Accordingly, real networks are expected to display different features. The creation of variants of the two models mentioned above is a theoretical reflection of variation in experiential measurements. Besides, depending upon which theoretical definition is chosen, the properties of both network models can be found in real world networks.

4.5 Statistical Properties of Networks

Determination of the statistical properties of a network is not only of mathematical interest, but also has practical implications. The information about the topological

structure of small-world networks, or scale-free networks, is expressed by using probability distributions. Therefore, considerable variation and randomness have been created at the individual level. For these models, there are no specific rules to follow, but the probability for certain actions might be higher than for others. For example, in a random graph, two nodes will not have a higher probability to link to each other because they have a common neighbour. In small-world networks, just the opposite is the case and a high clustering coefficient is a measure of this feature (Newman, 2000, Newman, 2001).

To summarise, in order to understand the nature of the mechanisms that lead to a certain network, it is useful to look at the topology of the finally emerging network. Are the links created randomly? Or are they created following specific rules? Is there a combination of randomness and certain rules? Another aspect to consider is that the networks are not only created differently, but they also behave differently depending on different topologies. All these questions should be carefully considered in order to understand the emergence of certain networks.

4.6 The Growth of Complex Networks

Random networks remain the main reference point for characterising complex networks. The classical random network is usually considered as a network with fixed number of nodes, and it is only the distribution of links between these nodes will be affected as the network grows. To be able to reproduce the features of real networks, we should consider the fact that the network itself could grow or decline (Albert and Barabási, 2002).

4.6.1 Exponentially Growing Networks

The simplest exponentially growing random network can be created as follows: one new node is added to the network in each time step. Then, another node is randomly chosen, without any preference, to be linked which the new node. Accordingly, the degree distribution of such growing network can be analytically expressed. It is expected to be different from random networks, but it will share with them the feature that the degree distribution for highly connected nodes will quickly decrease. Such networks are called exponentially growing networks (Dorogovtsev and Mendes, 2002).

An example of an exponentially growing network is a small-world network that begins with a fixed number of nodes connected in a regular order; see Figure 4.2. Then, two scenarios can be applied here: either a random rewiring of nodes with a certain probability (Watts-Strogatz model), or by adding links to a randomly chosen nodes with a certain probability (Newman-Watts model) (Wang, 2002).

4.6.2 The Preferential Attachment Mechanism

Scale-free networks are constructed differently from exponentially growing network. Not only the number of links grows, but also the number of nodes grows too. First, the network starts with a small number of nodes. Then, new nodes are inserted and connected to a certain number of already existing nodes. These nodes are selected with a probability proportional to the number of links k they have. Thus, the nodes that already have a large number of links are more likely to be selected to be linked to the new nodes. This process is described as *Preferential Attachment*, and it produces a scale-free network. The degree distribution of this network has a Power-

law tail, which is approximately equal to $k^{-\lambda}$, with $\lambda = 3$. This model is called the Barabási–Albert (BA) model.

According to Barabási, the combination of growth and preferential attachment is a simple model for producing a hierarchy. Thus, a node rich in links increases its connectivity faster than the rest of the nodes because incoming nodes link to it with higher probability (Barabási, 2001). The preferential attachment mechanism has also been called "popularity is attractive" (Dorogovtsev, et al., 2000). This is because if the linking behaviour is pointed toward the popularity or attractiveness of a node, this attractiveness can be expressed in terms of the number of links of a node.

As noted earlier, real networks usually express various degree distributions. Thus, detailed empirical investigations are required to find a good fit for the value of the exponent λ. The value of λ can differ among different real world networks, which indicate the presence of other mechanisms in addition to preferential attachment. In this context, new theoretical analysis emerged which gives different variants of the original BA model.

What seems to be crucial for the construction of scale-free networks is the type of dependency between the degree of a node and the probability of adding another link to it. The question is: what are the criteria for connecting a new node to already existing nodes? Consider a mathematical equation that expresses this relationship:

$$\Pi(k) = p + ak^{\alpha} \qquad (4.4)$$

The X(k) stands for the probability of adding a link to a certain node and depends on the number of links (degree) this node has (Albert & Barabási, 2002). Three parameters are present: p, a, and α.

The first term is simply a constant p, and it states the probability that a node will be connected, independently of the degree it has. If a node does not yet have links, there must also be a chance for such an isolated node to become connected within the growing network. This kind of basic wiring ensures that every node has a chance to become linked. Usually, it would be assumed that this probability is small. This mechanism is a simple but quite important extension of the original BA model. The way in which isolated nodes can be connected to a network has to do with the addition of innovation to a network. If a node that is already linked remained the only criterion for growth, then isolated nodes would never get a chance to be part of the network.

Once a node has a link, the second mechanism ak^{α} starts which ensures that the probability of connecting a node to other nodes will increase with the number of links that this node already has. This mechanism has been introduced previously as preferential attachment. The exponent α describes the way in which probability grows with degree. When $\alpha = 1$, a linear growth occurs and the value of λ will vary between 2 and ∞ depending on the other parameter a. If $\alpha < 1$, the degree distribution approaches a stretched exponential form. In the case that $\alpha > 1$, almost all nodes have a single edge, connecting them to a strongly connected node that has the rest of the edges of the network (Albert & Barabási, 2002).

To summarise, only when $\alpha = 1$ the scale-free character of the network is reconstructed, whereas any other form of preferential attachment seems to destroy the scale-free character of the network (Krapivsky, Redner, & Leyvraz, 2000). Accordingly, a node with a certain degree is always seeking for new nodes to increase its own degree. If only the degree of the node is relevant for the adding process, it is expected to have such a pure preferential attachment mechanism. If the neighbourhood of the node (e.g., the degree of neighbouring nodes) is also involved in the growth process, one would expect a non-linear growth rates with α unequal to one (Bruckner & Scharnhorst, 1986).

In the literature, further extensions of Equation 4.4 can be found by adding or modifying terms. Pennock et al. proposed a slightly different combination of two processes: the process of preferential attachment and the process of uniform attachment (Pennock et al., 2002). In addition, the process of introducing new links may be changed. If the number of these added links increase in time, then this is called *accelerated growth.* Nodes and links may have a limited lifetime, and networks that also decay can be considered. Dorogovtsev and Mendes found that the Power-law dependence in connectivity distributions remains if only a small fraction of links between old nodes are removed (Dorogovtsev & Mendes, 2000). The experimental investigation of these processes remains to be carried out via case studies. These would determine whether such processes occur in real-world networks, and what are the system-specific reasons for them to appear.

4.7 Dynamics in Complex Networks

Dynamics in networks refers to dynamic processes that take place in a network topology. An example of these dynamics is the robustness of the network against attack. One can think of the spread of computer viruses among computers. In this case, each node gets additional characteristics. It is "infected" or "not infected." The network topology defines the neighbourhood between nodes, and so the possible method of infection.

In particular, it has been shown that scale-free networks have a noticeable resilience to random connection failures, which implies that the network can resist a high level of damage (disconnected links), without loosing its global connectivity properties; i.e., the possibility to find a connected path between almost any two nodes in the system. (Pastor-Satorras & Vespignani, 2001).

Not all networks are equally vulnerable, and they may be more or less resilient in the face of different kinds of attacks. Holme et al. introduced different attack strategies by removing nodes or links. One strategy determines which objects are to be removed from the initial topological structure of the network (e.g., starting with the nodes with the highest degree). Another strategy recalculates the structure (e.g., ranking list of high degree nodes) after each step of removal (Holme et al., 2002). The difference in attack strategies showed the importance of changes in the network's structure during the attack. Recalculating strategies were the most effective for real networks.

Investigations of this kind become very important if the topology of networks is used to protect networks against attacks. Random networks are still the most robust networks. In addition, it has been recognised that scale-free networks are also relatively robust against random attacks, and that they can organise the flow of information effectively. Real world networks represent a combination between functionalities described by different models. The goal of analysis consists in understanding the mechanisms and driving forces behind these functionalities.

4.8 Summary

The last few years have witnessed the emergence of Complex Networks Theory. The development of this field started with experimental investigations of different types of real-world networks. It extended further into model-building activities that tried to mirror the statistical features found empirically. A rich class of different models has become available.

In this chapter, various network modelling efforts and the theoretical tools developed to model complex networks have been reviewed. Also a brief discussion of the three main classes of modelling complex networks is presented. First, the random graphs model which is highly used in many fields, as well as serve as a benchmark for many modelling and empirical studies. Then, the small-world model, which is located between highly clustered regular networks and random graphs. Finally, the scale-free models which is used to explain the origin of the Power-law tails and other non-Poisson degree distribution seen in real networks. Then, the topology of real world networks, mechanisms of growth, and the appearance of dynamic processes on these networks have been discussed.

Chapter 5

A STOCHASTIC LOAD-BALANCING ANALYSIS

Chapter 5

A STOCHASTIC LOAD-BALANCING ANALYSIS

5.1 Introduction

The analytical tools of Complex Network Theory have resulted in the creation of a number of dynamics and properties for complex networks that yield to predictable features. For example, a dynamic preferential attachment scheme provides a model for the evolution of developed networks (Newman and Park, 2003, Jin, el., 2001, Handcock and Jones, 2003, Lehmann, el., 2005), and explain why some networks have particular properties such as Power-law degree distributions and fault-tolerance to attacks and failures (Sarshar and Roychowdhury, 2004, Albert, el., 2000). Dynamics of such networks have led to efficient techniques for load distribution and search in peer-to-peer systems and power-law random networks (Sarshar, 2004).

In this thesis, and based on such studies, a dynamic network system is introduced such that the stationary degree distribution close to the degree distribution of ER random graphs. In ER random graphs, the probability of deviating from the average decreases exponentially with the deviation distance. The proposed load-balancing scheme is further improved to generate network system with stationary degree distribution close to that of regular graphs. I found that the network system studied here can provide an effective load-balancing mechanism for the distributed resources available on Grid Networks.

5.2 Load-Balancing Related Work

Future large-scale and Grid Networks are likely to be highly dense networks composed of thousands, hundreds of thousands, or even millions of nodes. Additionally, to contain the costs associated with deploying these networks, they will continue to be populated by low-cost, unreliable, power-limited nodes. As a consequence of this unreliability and the requirement to deploy these networks in harsh environments where partial destruction of the network may occur with high probability, new load-balancing and fault-tolerance algorithms should be designed to enhance the survivability of data collected by the network.

Consequently, there is a need for efficient, reliable, and scalable load-balancing paradigm for the distributed resources available on Grid Networks. Thus, when one node is overloaded, it can make use of unused computing power available in the network. Therefore, implementing an integrated load-balancing paradigm for an efficient load distribution and resource discovery will have an important role in implementing self-configuring and self-optimising networks, which are essential characteristics of Grid Networks.

Although several load-balancing algorithms for Grid Networks are proposed in the open literature, much of the analysis of these algorithms is limited to the results obtained from simulation; few have been studied using analytical methods and even fewer from measuring the performance of an actual Grid Network.

Let us consider a network system that consists of N nodes/resources. For efficient usage of these resources, it is desired to distribute the requests as evenly as possible,

so that no node is significantly more loaded than the others. However, this will not be simple to achieve for several reasons. The number of nodes is large causing a computational complexity in balancing the workload. Moreover, the uncontrolled dynamics of nodes where nodes have different capabilities and are geographically distributed with the possibility of loosing the connections between them at any time, makes the design of an efficient load-balancing algorithm a challenging task. Therefore, due to the above mentioned complexity considerations, it may be reasonable to introduce an element of randomisation in the problem formulation.

Load balancing is an active research field, and many methods and algorithms have been used to approach this problem (Drougas, et. al, 2006; Mitzenmacher, 2001; Peixoto, 1996, Mitzenmacher, 2001). The use of polling, agent-based methods, global random choice, randomised algorithms, and local diffusion methods have produced great advances in the field of load balancing (Murata, et al., 2006; Theimer and Lantz, 1989; Oppenheimer et al., 2004; Subramanian and Scherson, 1994; Els and Monien, 2003; Litzkow, Livny and Mutka, 1988; Bustos and Caromal, 2006; Yagoubi and Slimani, 2007).

However, most of these methods depend on *centralised* techniques, which can be efficient in small-scale networks, or on particular properties of the load distribution in larger networks. Furthermore, as central nodes require high computing power and large bandwidth, network systems that depend on such techniques are *Un-Scalable* (Lüling and Monien, 1993; Kremien & Kramer, 1992). Besides, *reliability* is another concern since the central server is a single point of failure.

In this thesis, a different load-balancing scheme is proposed. It is desired to create a dynamic network system that gives load distribution and resource updating without the need to monitor the nodes for their availability in a static network. Thus, no central nodes are required for resource allocation and load assignment. Moreover, in previous load balancing techniques, the structure of the network does not give information about the load status of the network; the network structure is used only to communicate load status information and to distribute the jobs, not to balance the load.

Some projects have been implemented to provide decentralised load balancing that use random walks on a network to distribute load (Montresor, et. al, 2002; Weiss, 1999). These projects are based on Multi-agent techniques, and load balancing is performed through a swarm of autonomous computing units, or agents, that travel randomly across the network trying to distribute the load. However, it is not clear how long the random walk will need to balance the load. In the proposed network system, it is aimed to reshape the network topology to successfully distribute the load and update resources via a limited number of the random steps. Furthermore, there is no analytical modelling for the load distribution in previous systems.

To address these issues, a statistical mechanic network system that provides a distributed load balancing mechanism is proposed. This network system has new contributions to distributed load balancing techniques since it will be *decentralised*, *self-organized*, *scalable*, and depends only on *local information* for load distribution and resource update. Furthermore, since there is no central point of failure and the networks created are based on random graphs, there is a higher possibility to find a

connected path between almost any two nodes in the system. This implies that the generated networks will be more *immune* to random errors, thus offering reliability to the network.

Based on jobs' arrival/departure rate equation analysis, it is shown that the steady state degree distribution corresponds to ER random graphs. It has been demonstrated that providing a local and distributed stochastic protocol, based on random sampling, can effectively enable the network to self-organise itself into ER random graphs. Extensive simulation results are presented which prove that the convergence of the generated network system into ER random graphs is robust even if the protocols have been modified in different ways to match practical implementation requirements. Therefore, the network system considered here can provide an effective load-balancing scheme for the distributed resources available on large-scale networks.

5.3 Proposed Load-Balancing Mechanism

The load-balancing technique proposed here can provide effective load balancing solution for Open Source software projects, Grid Networks, and other organisations seeking non-commercial load distribution solutions. Such network systems have hundreds of computing nodes and resources that are chosen randomly by users. Thus, the demand difference of the nodes can be quite large. Therefore, if each of the nodes can automatically redirect traffic to less loaded nodes, the nodes would have a more predictable load, and the users would have a more reliable service.

To effectively utilise nodes' resources, it is desirable to distribute the load requests

uniformly between the nodes, so that no node is significantly more loaded than others. Many load-balancing algorithms based on monitoring have been proposed (Wolf and Yu, 2001, Cardellini, el., 2002, Andreolini, el., 2002) and several open source projects have been created to give load balancing based on nodes' power and their geographical position (Horman, 2001, Horman, 2005, Zhang, 2003).

In this thesis, a load-balancing scheme that is different from those proposed in the literature is presented. Instead of monitoring the nodes and their available resources through a static network, a dynamic network system that provides a measure of instant load distribution status, and gives dynamics for job allocation and resource update is created.

To implement the proposed dynamic network system, a node's in-degree (node capacity) is mapped to its free resources or to the computation power of a node. Then, the edge dynamics are used to make the job allocation and resource maintenance procedures required for the load-balancing scheme. Accordingly, when a node receives a new job, it will remove one of its incoming edges to decrease its in-degree and indicate that its available resources are reduced. In the same way, when the node finishes a job, it will add an edge to itself to increase its in-degree. The process of adding and removing incoming edges is done by random sampling. In steady state, the rate at which jobs arrive would equal the rate at which jobs are finished and the network would have a fixed average number of edges. Therefore, a dynamic network system connecting all the nodes is developed.

The state of the network refers to the distribution of the jobs between the nodes. In

ideal conditions, the network system is expected to have the expected number of edges, and to have a binomial degree distribution close to that in ER random graphs.

My proposal is to correlate resources with in-degree and then allow the graph to reshape itself to reach the required edge dynamic. The nodes' in-degree distribution state in this network reflects the current distribution of load on the nodes. The job assignment and the resource update processes are achieved according to the edges' addition and deletion dynamics in the network. This will assure that the distribution of jobs will be relatively equal across all the nodes in the network. The in-degree distribution of the proposed network system will be similar to ER random graphs with a binomial degree distribution, which gives us load balancing. The proposed load-balancing protocol is described in details in the following sections of this chapter.

The stationary solution for the load distribution has been derived in this thesis. It is shown that this system can provide an effective load-balancing paradigm for the distributed resources accessible on Grid Networks.

As discussed in Section 4.2 of Chapter 4, the probability that an ER random graph has more or less than the expected number of edges decreases exponentially. This binomial distribution implies that each node will have a degree, which is close to the average degree, and that the number of nodes with much higher or much lower degree than average is very small. Thus, the probability that any node has the expected number of edges is the same.

Therefore, it is desired to develop a network system with degree distribution being converged around the average value. Thus, a network system equivalent to ER random graphs that are binomially distributed is needed to be defined, where each node will have the same probability to have the average number of edges, which will give us the required load balancing.

5.4 Stochastic Network System Modelling

The proposed load balancing algorithm creates a correlation between node's in-degree and its free resources. This implies that the network is optimally balanced and most of the nodes in the network must have the same in-degree and thus has the same free resources. Since ER random graphs have such dynamic, it is desired to reshape the network system to become close to ER random graph to achieve the required load balancing performance. Therefore, it is desired to generate a network system with degree distribution equivalent to those of ER random graphs that are binomially distributed.

In order to design such a dynamic system, the in-degree distribution of the nodes in a stochastic network system is analysed with a fixed number of nodes and fixed average number of edges. A node's in-degree refers to the free resources of a node. The job assignment and resource maintenance processes required for load-balancing are inherent in the network structure. Therefore, when a node receives a new job, it will remove one of its edges to decrease its in-degree. Similarly, when the node completes a job, it will add an edge to itself to increase its in-degree. The state of the proposed system refers to the distribution of the jobs between the nodes.

The increment and decrement of node's in-degree is performed via *Random Sampling*. Random sampling is the process whereby the nodes in the network are randomly picked up with equal probability. The random sampling will start at some fixed node, and at each step, it moves to a neighbouring node chosen randomly according to an arbitrary distribution. Then, in the last node in the walk of the random sampling will be selected for the load assignment. That is, when a node initiates a new job, a random sampling will be started through the network to assign the new job to the last node in the sampling. Then, the node who receives the new job will randomly remove one of its incoming edges to show that its load increased and its free resources decreased; see Figure 5.1.

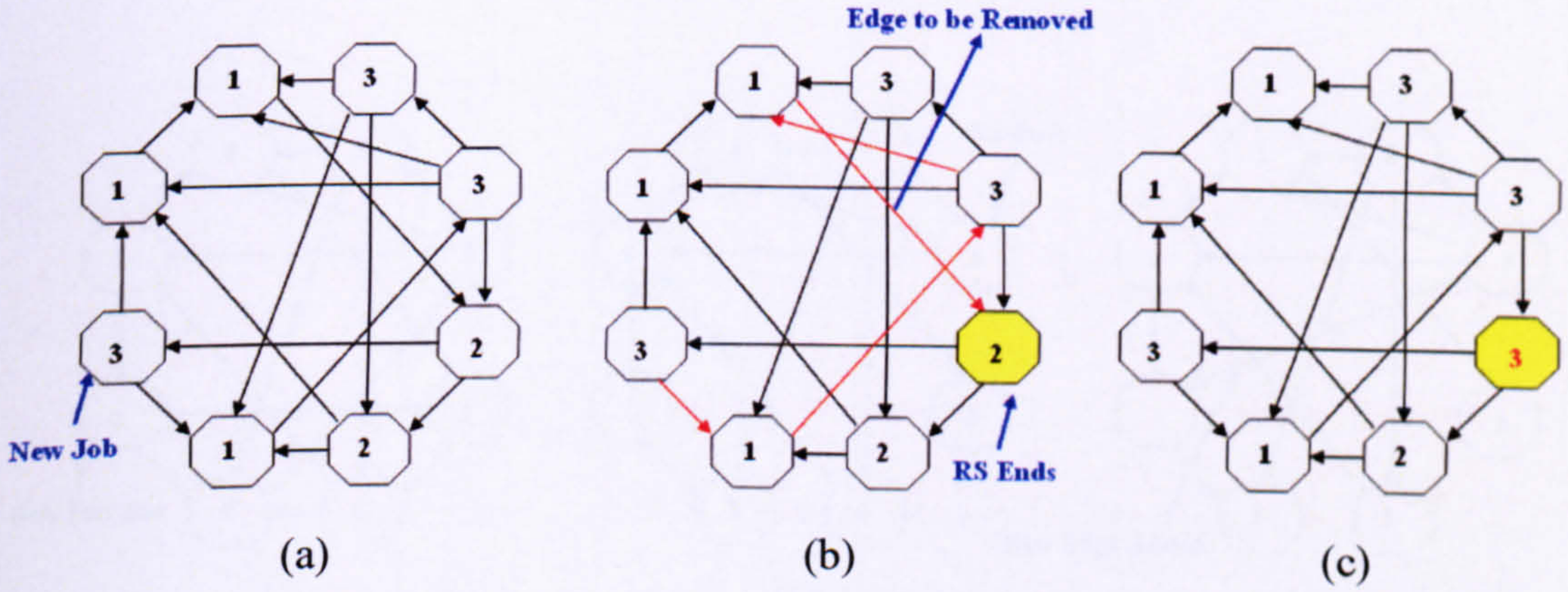


Figure 5.1 The random sampling procedure when a new job arrives.

In Figure 5.1, the large nodes represent the computing nodes in the network, and the label for each of the computing nodes indicates the current number of jobs it is running. Part (a) shows new job created in the network. In part (b), we see that the node who creates this new job initiates a random sampling. The last node in the sampling is selected to run the new job. To compensate for the additional job, the node which accepted the new job removes one of its incoming edges to show that its free resources decreased. The resulting network is represented in part (c).

Similarly, when a node finishes executing a job, it initiates a random sampling through the network and a new edge will be created to connect it to the last node in the sampling. This new edge will be added to the in-degree of the node that executed the job to show that its load decreased and its free resources increased.

Figure 5.2 shows the random sampling procedure when a running job finishes. Part (a) shows a running job finishing. In part (b) the node where job finishes initiates a random sampling through the network. In part (c) the last node on the sampling will be the origin of a new edge to connect it to the node that finished executing the job. This new edge represents the increase in available resources on the node where the job just executed.

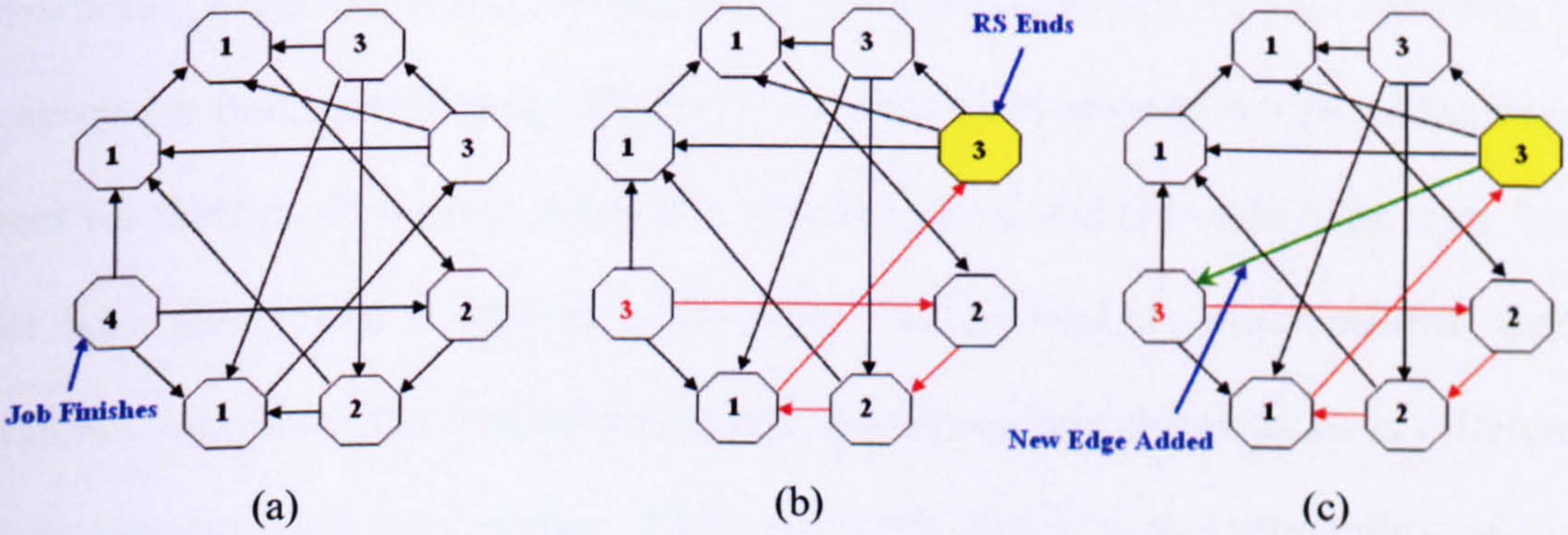


Figure 5.2 The random sampling procedure when a running job finishes.

The node's edge insertion/deletion process described above models the load assignment in the proposed dynamic load balancing algorithm, and the amount of free resources available for the nodes will show the job distribution status of the network. Therefore, node's edges are added or removed to keep the in-degree of a node proportional to its free resources.

As a result, the generated graph using this protocol will be a directed graph. For a directed graph with N nodes, each node has $(N-1)$ possible incoming edges, each of which is selected independently and with a definite probability. In addition, the direction of the edges in the directed graph is used to lead the propagation direction for the random sampling.

Analogous techniques have been used for load balancing which have produced some interesting results (Servetto and Barrenechea, 2002, Tian, el., 2005, Montresor, el., 2002, Avin and Brito, 2004). In (Servetto and Barrenechea, 2002), constrained random walk in Grid Network was studied. The constrained random walk algorithm chooses the next-hop neighbour through only the shortest path direction to forward the random walk. In the two directions of the shortest path, the probability of forwarding in each direction is recalculated in every step so that the load balancing is reached for multi-path routing. However, a regular Grid system on a plane has been used for their random walk, which is a special case of real networks. The work has not been generalised to random graphs embedded in arbitrary k-dimensional Grid systems. Moreover, the load-balancing algorithm proposed in this thesis is different from (Servetto and Barrenechea, 2002) since the direction and probability of the random sampling is not constrained in each step.

A random walk routing and load-balancing protocol in (Tian, el., 2005) is applied for Wireless Sensor Networks (WSNs). This random walk protocol does not require any location information, neither the exchange information between neighbouring nodes. However, this protocol is deployed only for specific WSNs applications where sensor nodes need to report their status to the base station (BS) from time to time by

sending a short message to the BS. The work is focused on the routing protocols for small size data transmission in WSNs with regular topologies to efficiently save energy and achieve long networking lifetime. In this contest, their protocol needed to be further improved to be extended to general WSNs, and to find other technique to prolong the network lifetime.

A different scheme is proposed here which has an advantage over the previous methods in that the network structure is dynamically changed to efficiently distribute the load. The proposed load-balancing paradigm will not require any monitoring mechanisms since it is encoded in the network structure. Moreover, the random sampling algorithm that will be used for nodes' selection will depend on the free recourses (in-degree) available for each node.

Load-balancing in general is not limited to the use of resources; there are other characteristics that can be used to effectively weight the sampled in-degrees, such as geographical distance, computing power, and available memory. Therefore, the random sampling can be biased in various ways to select certain nodes even if they are not the highest degree. The proposed load distribution scheme can utilise Grid resources performance monitoring toolkits to obtain runtime information such as computing power, processor available time, end-to-end network bandwidth, free memory, available resources, geographical location, and latency. Then, the proposed scheme calculates a non-uniform load distribution in the nodes based on all the above runtime information which will be used by the random sampling to select the target node that will receive the new load.

Therefore, the performance of the random sampling is improved by directing the sampling walk toward specific nodes (such as unvisited nodes, or nodes with certain properties) instead of choosing them unbiasedly at random. Furthermore, the number of sampling steps (or sampling length) will be limited to a finite length, and the nodes' selection criteria will be based on a predefined criteria rather than selecting the last node in the walk.

Random sampling is an uncontrolled process. The sampling walk can go to a neighbour and return to the same node after one step. This may result in some of the nodes being visited more times than the other nodes. However, if the network is regular and the random sampling is long enough, the steady state distribution of the random samplings will be uniform. This means that after a long enough time, the probability of the walk to be at any node is the same, which gives us *load balancing*.

In (Lov'asz and Winkler, 1995), the authors observed that in undirected graph, if the random walk was long enough, then in stationary state, the probability that the walk will stop at a specific node is proportional to its stationary in-degree distribution. This also can be applied to my directed graphs since the underlying network has fixed average number of edges. Hence, the generated graph using this technique will be a strongly connected directed graph. It is necessary that the graph be strongly-connected with no isolated components to ensure that the random sampling approach can effectively sample the graph. If many nodes in the network are isolated or if the network has multiple connected clusters, then the nodes will not be able to participate properly in the proposed load-balancing algorithm.

The connectedness of random graphs is well studied and it is an important measure of performance (see Albert and Barabási, 2002, and Chung and Lu, 2001). In order for a random graph to be a strongly connected graph, the average node's degree $\bar{k}$ should be greater than 3.5 (Chung and Lu, 2001). Therefore, the minimum average in-degree for the nodes is set to 4 to ensure that the network will be strongly connected network. It is not necessary that the graph must always have a single strongly connected cluster at every time step because the creation of new edges will heal the graph and make it a single strongly connected cluster again. This is an important practical aspect for an implementation and more detailed analysis will be the subject of future work.

ER random graphs have binomial distributions that exponentially decrease which provide good load-balancing. However, since every node in the network has the same in-degree, the optimal degree distribution is a regular graph as every node would have the same load. Modifications to the random sampling scheme to generate a more regular graph can further improve the load-balancing performance. Therefore, biasing the sampling toward specific nodes instead of choosing them uniformly randomly will improve the random sampling performance.

Accordingly, a biased random sampling (BRS) is proposed here to provide the required dynamic load-balancing and the edge insertion and deletion strategy assures that the load will be distributed equally across all the nodes in the network. The BRS load-balancing technique assigns the new job to the least loaded (or highest in-degree) node in the random sampling. The generated network system is expected to have in-degree distribution close to regular graphs. The main initiative of the BRS

load-balancing scheme is that the network structure can represent the load distribution status of the network.

Figure 5.3 shows the BRS procedure when a new job arrives. Part (a) shows new job initiated in the network. In part (b), the node who creates the new job initiates a random sampling which keeps track of the in-degree (or free resources) of each visited node. The highest in-degree node (most free resources) is selected to run the new job. Then, the node which accepted the new job removes one of its incoming edges to show that its free resources decreased. The resulted network is showed in part (c).

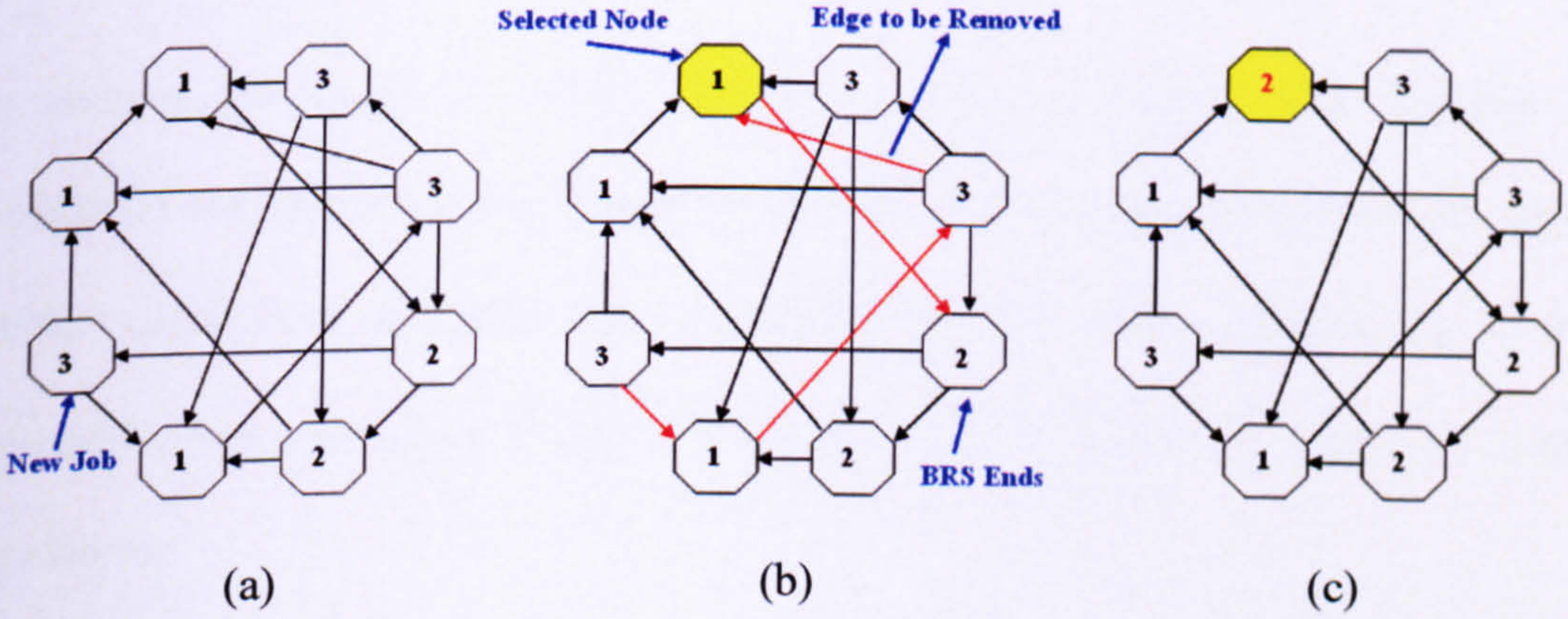


Figure 5.3 The biased random sampling (BRS) procedure when a new job arrives.

One key advantage of the BRS scheme is that the data structure required for load-balancing is dynamically retained by each node through the links (or edges) used to connect with other nodes in the network. The nodes in the network will create these edges when the load is distributed using random sampling and/or when edges are being inserted or deleted. These edges are vital because they hold the state of the network. Therefore, it is essential to keep the network in the proper state in order to

keep the BRS load-balancing scheme in the proper state to effectively distribute the load. Besides, the network state maintenance is easily attained using local edge connecting procedures. Therefore, if a link is failed, it can simply be fixed before it has an effect on the system performance.

5.5 Stationary In-Degree Distribution Analysis

The proposed load distribution mechanism is difficult to analyse mathematically due to the dynamic nature of the network and the way the random sampling works. Therefore, the analytical verification was simplified by restricting the load distribution mechanism to use a simple random sampling scheme, which selects the last node in the walk, rather than using the proposed random sampling scheme.

To analyse the generated network system, a network system with N nodes is considered for this work and it is assumed that all the nodes in the network have similar capabilities and jobs can be executed in any node. Suppose p_k is the probability that a node has k edges. Then, the average number of edges, E, in the network is:

$$E = N\sum kp_k \tag{5.1}$$

At each step, a randomly chosen edge will be deleted, and a randomly chosen edge will be inserted. Thus, the numbers of edges inserted and deleted are both random variables that are selected to have a fixed average number of edges.

Let D be the average number of deleted edges in the network, and let M be the

maximum number of edges a node can have. To make the network system compatible with ER random graphs, it is assumed that each node can have up to $(N-1)$ edges, thus $M \leq (N-1)$. This assumption is not a limitation of the mechanism; it is only to show that this system is designed for large-scale networks.

So, the expected number of edges in the network is given by:

$$E = NM - D \tag{5.2}$$

Since the probability that a random sampling with a proper length will stop at a specific node is proportional to its stationary in-degree, thus, if node's edges have been deleted uniformly randomly, then the probability that a node with k edges will lose one or more of its edges is proportional to its in-degree. Therefore, the rate at which the in-degree of a specific node with k edges will decrease is given by:

$$R_k = \frac{k}{E} = \frac{k}{NM - D} \quad , \quad 0 \leq k \leq M \tag{5.3}$$

Here, the node's in-degree will decrease only by one edge at a time.

In the same way, the probability that the in-degree of a certain node will increase is proportional to the number of deleted edges from this node. Thus, the node's in-degree will increase by one at a rate given by:

$$S_k = \frac{M - k}{D} \quad , \quad 0 \leq k \leq M \tag{5.4}$$

Since the average number of edges is assumed to be fixed, the network can be described as a Markov Chain (Kleinrock, 1975) with insertion and deletion rates given by Equations (5.3) and (5.4). In this Markov Chain, the node's in-degree represents Markov states with the probability of going from one state to another being given by R_k and S_k respectively by deletion or addition of an edge. Therefore, the total number of states in this Markov Chain is $(M+1)$, where they refer to the node's in-degree in this dynamic system, and the rates at which node's in-degree decreases and increases are given by R_k and S_k respectively.

For Markov Chain, the stationary distribution for the expected number of jobs per node (node's in-degree) is defined by the following expression:

$$V[\mathrm{A}-I]=0 \tag{5.5}$$

where A is the transition matrix, I is the identity matrix, and V is the distribution vector (transition probability) and it is defined as:

$$V=[p_{k+1} \quad p_k \quad p_{k-1}] \quad , \quad 1\le k<M \tag{5.6}$$

Figure 5.4 shows the transition graph (states) for node's in-degree. From Figure 5.4, the transition matrix T can be obtained;

$$T=\begin{bmatrix} 1-R_{k+1} & R_{k+1} & 0 \\ S_k & 1-(R_k+S_k) & R_k \\ 0 & S_{k-1} & 1-S_{k-1} \end{bmatrix} \tag{5.7}$$

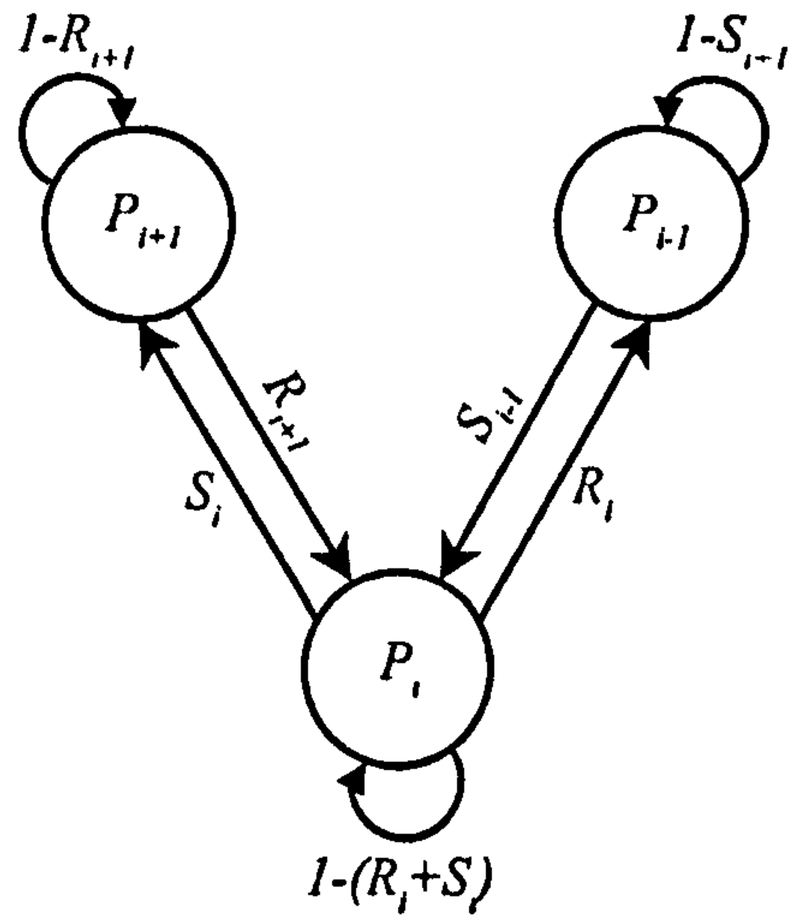


Figure 5.4 The transition graph (states) for node's in-degree in the network.

Please note that the in-degree of the node is decreased or increased by only one edge.

Using Equations (5.5), (5.6), and (5.7), I have

$$[p_{k+1} \quad p_k \quad p_{k-1}]\left[\begin{bmatrix} 1-R_{k+1} & R_{k+1} & 0 \\ S_k & 1-(R_k+S_k) & R_k \\ 0 & S_{k-1} & 1-S_{k-1} \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right] = 0 \qquad (5.8)$$

$$[p_{k+1} \quad p_k \quad p_{k-1}]\begin{bmatrix} -R_{k+1} & R_{k+1} & 0 \\ S_k & -(R_k+S_k) & R_k \\ 0 & S_{k-1} & -S_{k-1} \end{bmatrix} = 0 \qquad (5.9)$$

Then, by expanding Equation (5.9), it gives

$$-p_{k+1}R_{k+1} + p_k S_k = 0 \qquad (5.10)$$

$$p_{k+1}R_{k+1} - p_k(S_k + R_k) + p_{k-1}S_{k-1} = 0 \qquad (5.11)$$

$$p_k R_k - p_{k-1} S_{k-1} = 0 \tag{5.12}$$

From the above equations, we can see that in the steady state, the rate at which node's in-degree increases will equal the rate at which node's in-degree decreases. Therefore, the network system generated has a fixed expected number of edges.

From Equation (5.12), I have

$$p_k = \frac{S_{k-1}}{R_k} p_{k-1} \tag{5.13}$$

Now, if $k = 1$, then from Equation (5.13) I have

$$p_1 = \frac{S_0}{R_1} p_0 \tag{5.14}$$

Similarly, if $k = 2$, then

$$p_2 = \frac{S_1}{R_2} p_1 = \frac{S_1}{R_2} \cdot \frac{S_0}{R_1} p_0 \tag{5.15}$$

Thus, the probability that a node has k edges, p_k, becomes

$$p_k = \frac{S_{k-1}}{R_k} p_{k-1} = \frac{S_{k-1} S_{k-2} \cdots S_0}{R_k R_{k-1} \cdots R_1} p_0 \tag{5.16}$$

By inserting equations (5.3) and (5.4) into equation (5.16), it becomes

$$p_k = \frac{M(M-1)(M-2)\cdots(M-(k-1))D^{-k}}{k!(NM-D)^{-k}} p_0 = \binom{M}{k}\left(\frac{NM-D}{D}\right)^k p_0 \qquad (5.17)$$

Moreover, since the total probability P_k is equal to one, then I have

$$P_k = \sum_k p_k = \sum_k \binom{M}{k}\left(\frac{NM-D}{D}\right)^k p_0 = 1 \qquad (5.18)$$

By using the Binomial Expansion Theorem (Abramowitz and Stegun, 1972), I have

$$p_0 = \left(\frac{D}{NM}\right)^M \qquad (5.19)$$

Thus, after simplifying the equations (5.16) through (5.19), we can see that p_k is binomially distributed and it is given by

$$\begin{aligned} p_k &= \binom{M}{k}\left(\frac{NM-D}{D}\right)^k\left(\frac{D}{NM}\right)^M = \binom{M}{k}\left(\frac{NM-D}{NM}\right)^k\left(\frac{D}{NM}\right)^{(M-k)} \\ &= \binom{M}{k}\left(1-\frac{D}{NM}\right)^k\left(\frac{D}{NM}\right)^{(M-k)} \end{aligned} \qquad (5.20)$$

This degree distribution implies that the proposed network system is equivalent to the degree distribution of ER random network as illustrated in Equation (4.2) in Chapter 4.

$$p_k = \binom{M}{k} q^k (1-q)^{(M-k)} \tag{5.21}$$

Thus, for each node, there are M possible incoming edges, each of which is selected independently and with connection probability C_p given by

$$C_p = 1 - \frac{D}{NM} = \frac{NM - D}{NM} = \frac{E}{NM} \tag{5.22}$$

These analytical results show that the stationary distribution is compatible with ER random networks. Therefore, if the number of edges in the network is fixed with connection probability $C_p = E/(NM)$, it is expected that each node in the network will have E edges, which gives us *load-balancing*. Thus, the proposed algorithm gives nearly optimal load-balancing performance by creating an almost regular network system where each node's in-degree refers to its free resources.

5.6 The BRS Load-Balancing Scheme Implementation

The proposed BRS load-balancing scheme is an easily implemental scheme using standard networking protocols. The decentralised feature of the scheme makes it suitable for many large network systems such as Grid Networks. Hence, we can apply the proposed load-balancing scheme on top of Grid Network as a virtual network (Adabala et al., 2005), or, we can integrate it inside Grid Network Middleware (Schantz and Schmidt, 2001; Blair et al., 1999). For example, this network system can be built directly on top of any of the physical transport layers and use the Grid Network as its underlying network.

Grid Networks consist of computers and routers, which are connected by different physical links (Ethernet, Serial, ATM, Wireless, etc.). By using a virtual IP addresses, networked computers can be addressed without the knowledge of the physical transport layer. Therefore, the network does not need to consist of physical connections between nodes. Thus, in an addressable system, the nodes' edges may simply be a table of IP addresses that each node constructs to represent its neighbours in the network. If the networks are not globally addressable, each node will need to maintain a routing table that gives complete route information on how to reach its neighbours, or the actual physical links. Accordingly, network edges can be used to represent cached routes in the underlying physical layer and lightweight and fast transport protocols (e.g. User Datagram Protocol (UDP), Lightweight Directory Access Protocol (LDAP), Lightweight TCP/IP (lwIP)) can be used to represent the large number of edges of the network with minimum overhead. Each node will have local information about its status (i.e. its free resources available), which can be used for resource allocation and load distribution.

To implement the proposed algorithm and to generate the network dynamics analysed in the previous sections, there is a need for a random dynamic that samples the network using local information to choose which edges to insert or remove. As can be observed from Equating (5.3), the nodes lose their edges preferentially with respect to their in-degree. And since in steady state and when the random sampling is long enough, the probability that the random sampling will end at a particular node is proportional to its in-degree. Therefore, the proposed random sampling technique can be used to implement the desired graph dynamics which will sample the network using only local information to efficiently distribute the jobs among the nodes.

The performance of proposed load balancing technique has been improved by assigning the new job to the least loaded (or highest in-degree) node in the sampling, instead of the last node in the walk. When the node with the most-free resources on a walk is preferred to receive the new job, its resources must be greater than or equal to the resource of the last node on the random sampling. Hence, it is expected that improved load balancing technique will have the same scalability as the standard random sampling, and the balancing performance is much improved as shown in simulation results in Chapter 6.

In addition, the random sampling is further improved by directing the sampling toward specific nodes (such as unvisited nodes, or nodes with certain properties) instead of choosing them uniformly at random. Hence, the nodes' selection criteria will be based on a predefined criterion rather than selecting only the last node in the walk, and the length of sampling walk will be limited to a finite length.

5.7 Network Simulation Methodology

Simulations are used to evaluate the performance of load-balancing algorithm and to show that the resulted dynamics match the theoretical predictions. For network simulations, a network system is created which has N nodes. Each node in the network is a computer with computing power equal to its maximum in-degree. One unit of power can process a unit of load in each unit of time. The number of edges in each node will be proportional to its free resources. Simulation timing (iteration) is the time required to send a message or a data packet from one node to another node.

Initially, experiments were carried out under ideal conditions where it is assumed

that all the nodes in the network have similar capabilities and jobs can be executed in any node. Such assumption was made to prove the concept of the load-balancing scheme and to verify that the proposed network system certainly leads to ER random networks and matches the analytical results. Here, the random sampling will select the last node in the sampling to match the predicted theoretical analysis. After that, the proposed scheme has been modified to suit more real networks by adding heterogeneity to the nodes and biasing the random sampling toward specific nodes.

In a homogeneous configuration, all nodes have equal capabilities and jobs can run on any node. For homogeneous system simulations, the number of jobs that arrive and depart each time step is Poisson distributed, and the average job arrival rate is 512 jobs at each time step. In addition, the job size is Poisson distributed and the average job size is 512 Kbits. Applying Poisson distribution in homogenous system simulations is discussed in more details in Chapter 6.

In a heterogeneous configuration, the nodes have different capabilities and varying resources. Each node in the network has an in-degree equal to its computational power (capability). For heterogeneous system simulations, Pareto (or Power-law) distribution (Mitzenmacher, 2003) has been used to model nodes' computation power distribution, job's arrival rate and job sizes. These simulations can give a practical reflection of how the system would work under real situations. Table 5.1 summarises the simulation parameters for heterogeneous system simulations. Pareto distribution is discussed in more details in Chapter 6.

Table 5.1 Parameters for heterogeneous system simulations.

In-Degree Distribution		Job Arrival Rate		Job Size (Kbits)	
Minimum	Maximum	Minimum	Maximum	Minimum	Maximum
4	100	1	4096	32	2048

The random sampling protocol used for load distribution depends only on local information (e.g. node's available resources, communication delay, computing power, and available memory) to sample the network and to efficiently distribute the workload among the nodes. The length of random sampling will be limited to a specific number of steps which will vary according to experiment's requirements.

In simulations, the random sampling algorithm assigns the new jobs to the sampled nodes according to a predefined criterion from the following:

- The last node in the random sampling to compare the results with the mathematical predictions.
- The node which has the highest in-degree (most free resources) in the sampling to efficiently balance the load distribution.
- The node with least communication delay to reduce the average communication latency in the network.

To examine the performance of the proposed load-balancing scheme in practical networks, two types of simulation experiments were carried out. The first experiment considered the computing power alone as the key factor for load balancing. In the second experiment, the geographical distance (communication delay) is added as a second factor for load balancing. Accordingly, the proposed load-balancing scheme

models the network as a graph consisting of weighted nodes and communication edges. Therefore, the load balancing scheme objective is to balance the load between the nodes as well as to minimise the communication latency.

In all of these simulations, we begin with a randomly-connected network subject to the initial degree distribution. The node's edges are added or removed to keep the in-degree of a node proportional to its free resources. Hence, when a node receives a new job, it randomly removes one of its incoming edges to show that its load has increased and its free resources have decreased. Similarly, when a job is completed, the in-degree of the node that executed the job will be increased to show that its load has decreased and its free resources have increased. This is done by randomly sampling the network, and then, a new directed edge will be created from the last node in the sampling to the node that executed the job.

The generated network is a directed graph and the direction of edges is used to lead the propagation direction for the random sampling. Every node in the network must keep the minimum in-degree so that the network remains strongly-connected. Therefore, the minimum in-degree for the nodes is set to 4 to guarantee that the network will be a strongly-connected network.

The node's edge insertion and deletion process described above will simulate the change in the workload of the network, and the amount of free resources available for the nodes will exhibit the job distribution status of the network. The state of the network represents the instantaneous distribution of jobs over all the nodes.

5.8 Simulation Description

The following steps were used to simulate the network and job traffic on the network and to find out whether or not a regular graph could develop from random initial topologies:

- **Graph Initialisation:** First we create a directed graph with N nodes and the maximum degree of any node is M. The nodes have in-degree proportional to their free resources. The free resources are defined to be the amount of computing power available. The initial structure of the network is created by connecting each node in the network to a random number of nodes. This graph is intentionally constructed in a random manner to show that the proposed algorithm lead to a regular graph independent of the initial configuration.

- **Edge Deletion (Job Submission) Process:** When a new job is submitted, a random sampling of length $\log(N)$ is started from the node that submitted the new job. In standard random sampling (RS) algorithm, the last node on the random sampling will be selected to receive the new job. Figure 5.5 shows the pseudo code for procedure SelectDestinationRS(source).

Procedure SelectDestinationRS(source) – when a new job enters the network, a random sampling initiated to select the last node in sampling to give it the new job.

```
1: RS_length ← log(N), steps ← 0
2: destination ← source
3: while (steps < RS_length)
5: destination ← RandomOutNeighbor(destination)
6: steps ← steps + 1
7: end while
8: return destination
```

Figure 5.5 The pseudo code for procedure SelectDestinationBRS(source).

In the biased random sampling (BRS) algorithm, the highest in-degree node on the sampling will be selected to receive the new job rather than the last node to improve the efficiency of the proposed BRS load-balance algorithm and generate networks closer to regular graphs. Therefore, when a new job enters the network, a random sampling is initiated that retain information about the nodes' free resources during the random sampling. Then, the job is assigned to that highest in-degree node. Figure 5.6 shows the pseudo code for procedure SelectDestinationBRS(source).

Procedure SelectDestinationBRS(source) – when a new job enters the network, a random sampling is initiated that retain information about the nodes on the sampling. The job is then assigned to that highest in-degree node.

```
1: BRS_length ← log(N), node ← source, steps ← 0
2: indegree ← node.resources, destination ← node
3: while (steps < BRS_length)
4: node ← RandomOutEdge(node)
5: indegreetemp ← node.resources
6: if (indegreetemp > indegree) then
7: indegree ← indegreetemp
8: destination ← node
9: end if
10: steps ← steps + 1
11: end while
12: return destination
```

Figure 5.6 The pseudo code for procedure SelectDestinationBRS(source).

Accordingly, when the node receives the new job, it uniformly randomly deletes one of its incoming edges. These edges are deleted to reflect the decrement of nodes' free recourses; see Figure 5.7 for the pseudo code for procedure DeleteIncomingEdge(destination).

Procedure DeleteIncomingEdge(destination) – when an node receives a new job, it deletes on of its incoming edges to reflect the decrement of its free recourses.

```
1: source ← destination
2: source ← RandomInEdge(destination)
3: DeleteEdge(source, destination)
4: destination.resources ← destination.resources – 1
5: end
```

Figure 5.7 The pseudo code for procedure DeleteIncomingEdge(destination).

– **Edge Addition (Job Finishing) Process**: The addition of an edge is undertaken when a node finishes one of its jobs and wants to increase its in-degree. Thus, when a node completes a running job, it initiates a random sampling of length $\log(N)$ through the network and the last node in sampling to be connected to the node that finished the job. Figure 5.8 shows the pseudo code for procedure SelectSource(destination). Then, a new directed edge will be added from the last node that the sampling ends at to the node which finished the job to account for its increase in free resources. Figure 5.9 shows the pseudo code for procedure AddIncomingEdge(source, destination).

Procedure SelectSource(destination) – when a node finishes a job, it initiates a random sampling that selects the last node in sampling to be connected with the node that finished the job.

```
1: RS_length ← log(N), steps ← 0
2: source ← destination
3: while (steps < RS_length)
4: source ← RandomOutEdge(source)
5: steps ← steps + 1
6: end while
7: return source
```

Figure 5.8 The pseudo code for procedure SelectSource(destination).

Procedure AddIncomingEdge(source, destination) – when a node finishes a job, a new directed edge will be added from the last node in the random sampling to the node that finished the job to show that its free resources is increased.

```
1: AddEdge(source, destination)
2: source.resources ← source.resources + 1
3: end
```

Figure 5.9 The pseudo code for procedure AddIncomingEdge(source, destination).

5.9 Summary

In this chapter, a network system that gives a distributed load-balancing scheme by generating almost regular graphs is proposed. This network system is self-organized and depends only on local information for load distribution and resource discovery. The in-degree of each node refers to its free resources, and job assignment and resource updating processes required for load balancing is accomplished by using random sampling. The idea is to correlate resources with in-degree and then allow the graph to reshape itself to reach the required edge dynamic. An analytical solution for the stationary degree distributions has been derived which confirms that the edge distribution of proposed network system is compatible with ER random graphs. Thus, this network system can provide an effective load-balancing paradigm for the distributed resources accessible on Grid Networks.

Simulations description and the key procedures and parameters that have been used to conduct the network simulations are summarised in this chapter. Extensive simulations were performed to validate the efficiency and scalability of the proposed load-balancing mechanism and the results are reported in this and the following chapter.

Chapter 6

PERFORMANCE EVALUATION, RESULTS, AND DISCUSSION

Chapter 6

PERFORMANCE EVALUATION, RESULTS, AND DISCUSSION

6.1 Introduction

In order to evaluate the proposed load-balancing scheme, and to verify that the proposed network system generates almost regular graphs and matches the analytical results. Extensive simulations are performed with various parameters and the results are reported in this chapter.

To verify the efficiency and scalability of the proposed load-balancing mechanism under various conditions, few adjustments to the developed system have been done. For instance, the network size does not have to be constant and the random sampling will be biased to preferentially select the nodes according to specific conditions. Moreover, the performance of my load balancing technique under heterogeneous configurations is examined where the nodes in the network have different capabilities and resources. Furthermore, the affect of communication latency on load distribution and nodes' in-degree is studied. The steady state in-degree distribution, the in-degree standard deviation (or variance), and the correlation have been used to assess the load-balancing performance.

Simulation results have been used to evaluate and examine the performance of the

load-balancing algorithm and to determine the optimum random sampling length required to achieve the required load balancing. Then, the scalability and reliability of the algorithm is evaluated under several conditions. The effect of including localisation information to the random sampling on the average communication latency and the load-balancing performance is examined.

6.2 Load Balancing Performance vs. Theoretical Predictions

To study the performance of the proposed load-balancing scheme, the in-degree distribution of the generated network by using the *random sampling* (RS) protocol is examined and compared with the *predicted* binomial in-degree distribution. Here, the available resources (e.g. computing power) are considered as the key factor for selecting the node which will receive the new load. Moreover, the random sampling protocol used here will select the *last* node in the walk to compare simulation results with theoretical results.

Here, the performance of proposed load balancing mechanism for homogeneous network systems is examined, where all nodes have equal capabilities and jobs can run on any node. Such assumption is made to prove the concept of load-balancing scheme and to verify that the proposed network system certainly leads to ER random networks and matches the analytical results.

In homogeneous system experiments, the number of jobs that arrive and depart each time step is Poisson distributed, and the expected job arrival rate, $\bar{\nu}$, is set to 512 jobs at each time step. Also, the size of each job is Poisson distributed and the

average job size, $\overline{\beta}$, is set to 512 Kbits.

Figures 6.1 and 6.2 show that the steady state in-degree distribution for the network is very close to the binomial distribution described in Section 4.1 in Chapter 4. The network under consideration in these simulations has $N = 512$ nodes, and the maximum in-degree $M = N - 1$. The node's average in-degree $\overline{k}$ for the network whose simulation is shown in Figure 6.1 is 32 and for that in Figure 6.2 is 72. Recall that predicted steady state in-degree distribution $P(k)$ is calculated according to the following equation (Erdös and Rényi, 1960, Bollabás, 2001):

$$P(k) = e^{-pN} \frac{(pN)^k}{k!} = e^{-\overline{k}} \frac{(\overline{k})^k}{k!} \qquad (6.1)$$

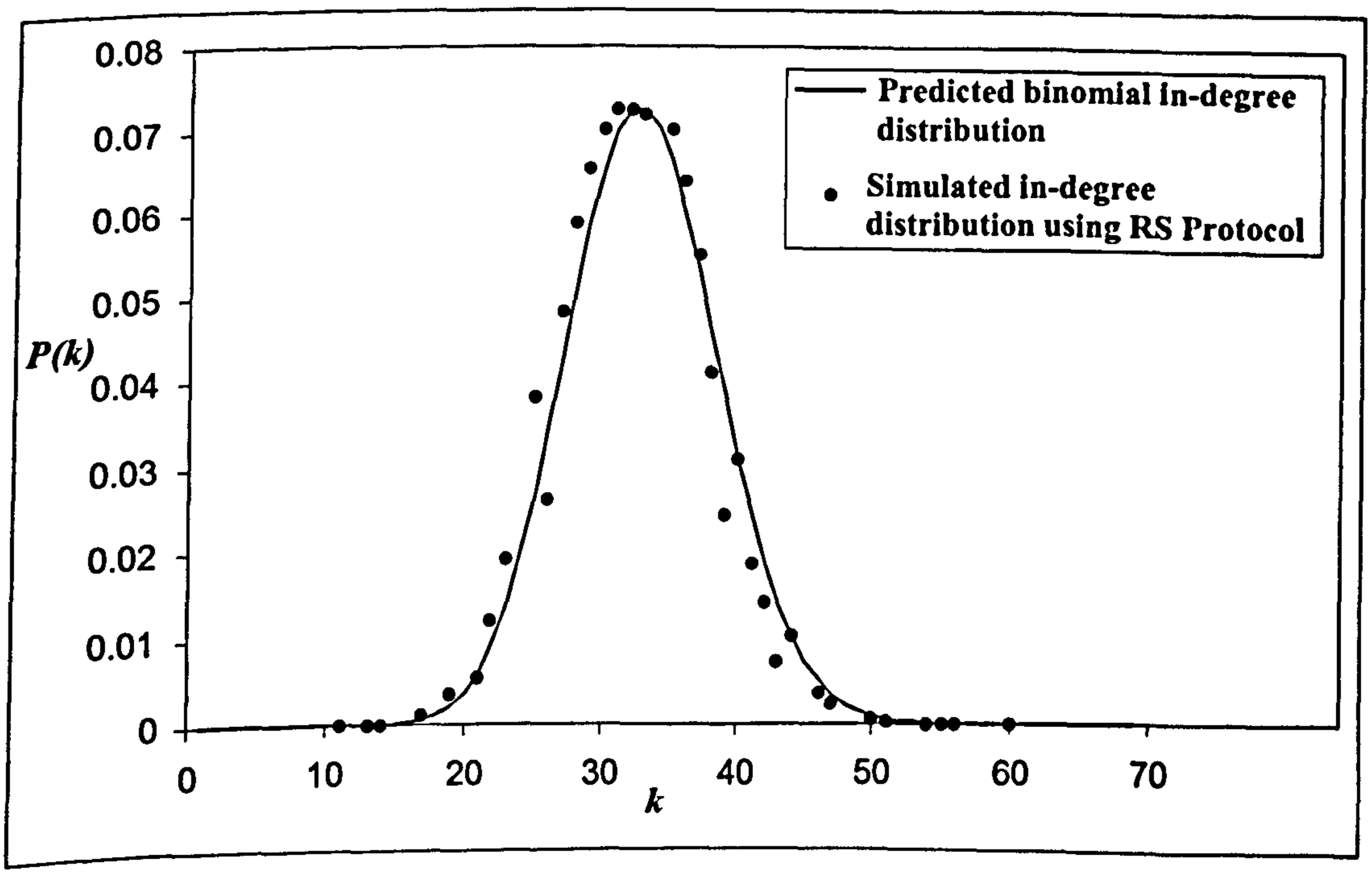


Figure 6.1 The simulated steady state in-degree distributions using random sampling (RS) protocol compared with the predicted binomial distribution for networks with node's average in-degree $\overline{k} = 32$ and network size $N = 512$.

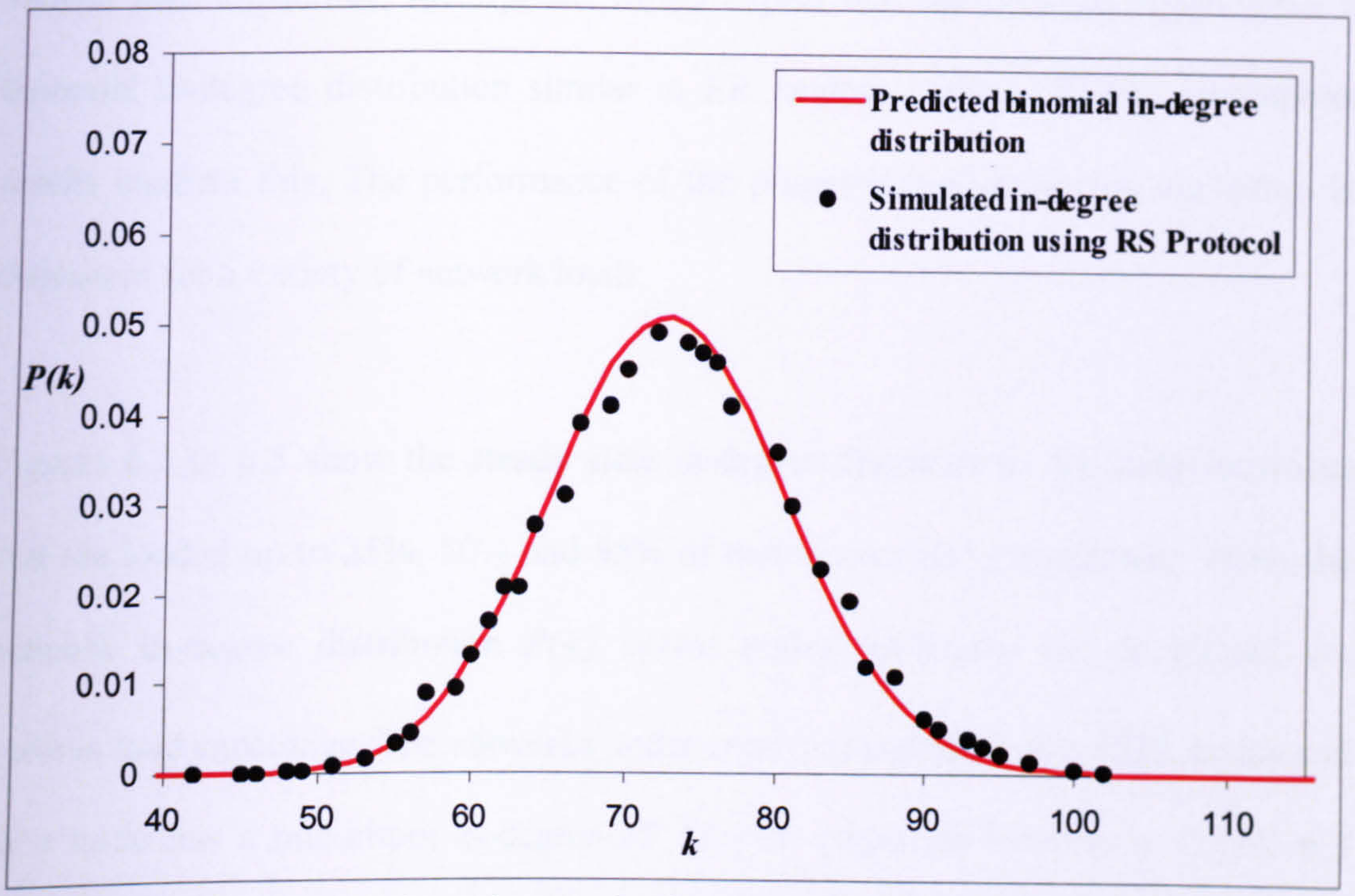


Figure 6.2 The simulated steady state in-degree distributions using random sampling (RS) protocol compared with the predicted binomial distribution for networks with node's average in-degree $\overline{k} = 72$ and network size $N = 512$.

Simulation results confirm that the proposed network dynamic creates ER random networks, and the proposed random sampling technique can be used to efficiently distribute the load between the nodes. The correlation created by the load-balancing technique between a node's in-degree and its free resources implies that most of the nodes in ER random graph have the same in-degree and the same free resources. Thus, a nearly optimal load balancing performance is achieved.

6.3 Load-Balancing Performance under Different Network Loads

Simulations have been extended to analyse the proposed load-balancing technique under several parameters and conditions. For example, to study the performance of the algorithm under different network loads, the network has been examined under

various load capacities. Ideally, we would expect that the network would have a binomial in-degree distribution similar to ER random graphs. Indeed, simulations results confirm this. The performance of the proposed load-balancing algorithm is consistent for a variety of network loads.

Figures 6.3 to 6.5 show the steady state in-degree distributions for three networks that are loaded up to 25%, 50% and 85% of their capacities; respectively. Here, the network in-degree distribution $P(k)$ versus nodes' in-degree (k) is plotted for various load capacities. The networks under consideration have $N = 1024$ nodes and each node has a maximum in-degree of $M = 64$ edges (or resources). Figure 6.3 shows a network that is lightly loaded with less than 25% of its capacity. Figure 6.4 shows a network that is about 50% loaded. Finally, figure 6.5 shows a network that is heavily loaded to about 85% of its capacity.

Ideally, it is expected that the network would have a binomial in-degree distribution similar to ER random graphs. Indeed, simulations results confirm this. As we can see from the figures, the networks experiencing a large range of network loads benefit from nearly equivalent load-balancing performance. Thus, whether the network is heavily loaded or nearly idle, the load-balancing performance on the network is almost the same and the resulting in-degree distribution is close to ER random graph. Thus, the proposed load-balancing algorithm is effective for different network loads.

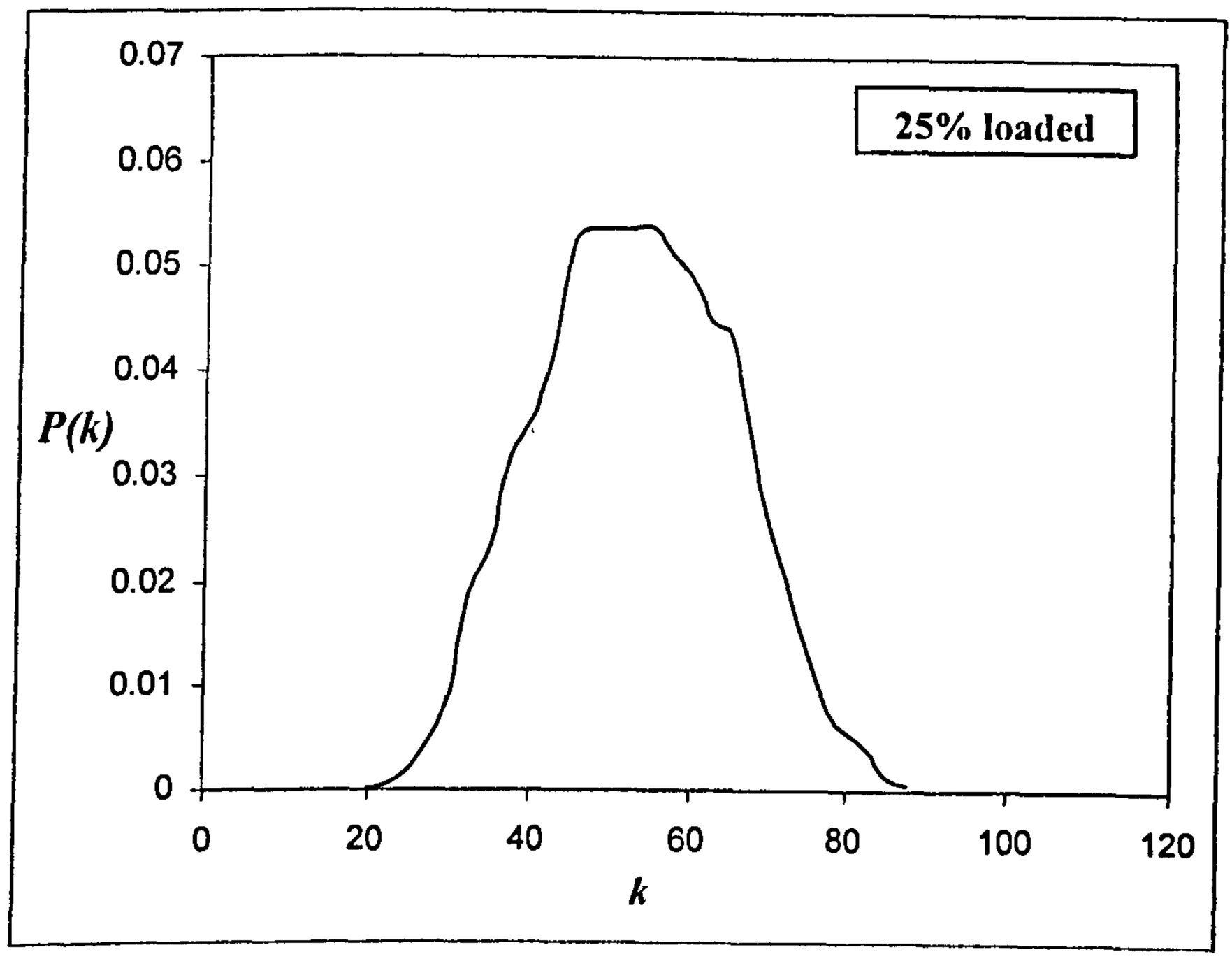


Figure 6.3 The in-degree distributions when the network is loaded up to 25% of its capacity with *N=1024* and *M= 64*.

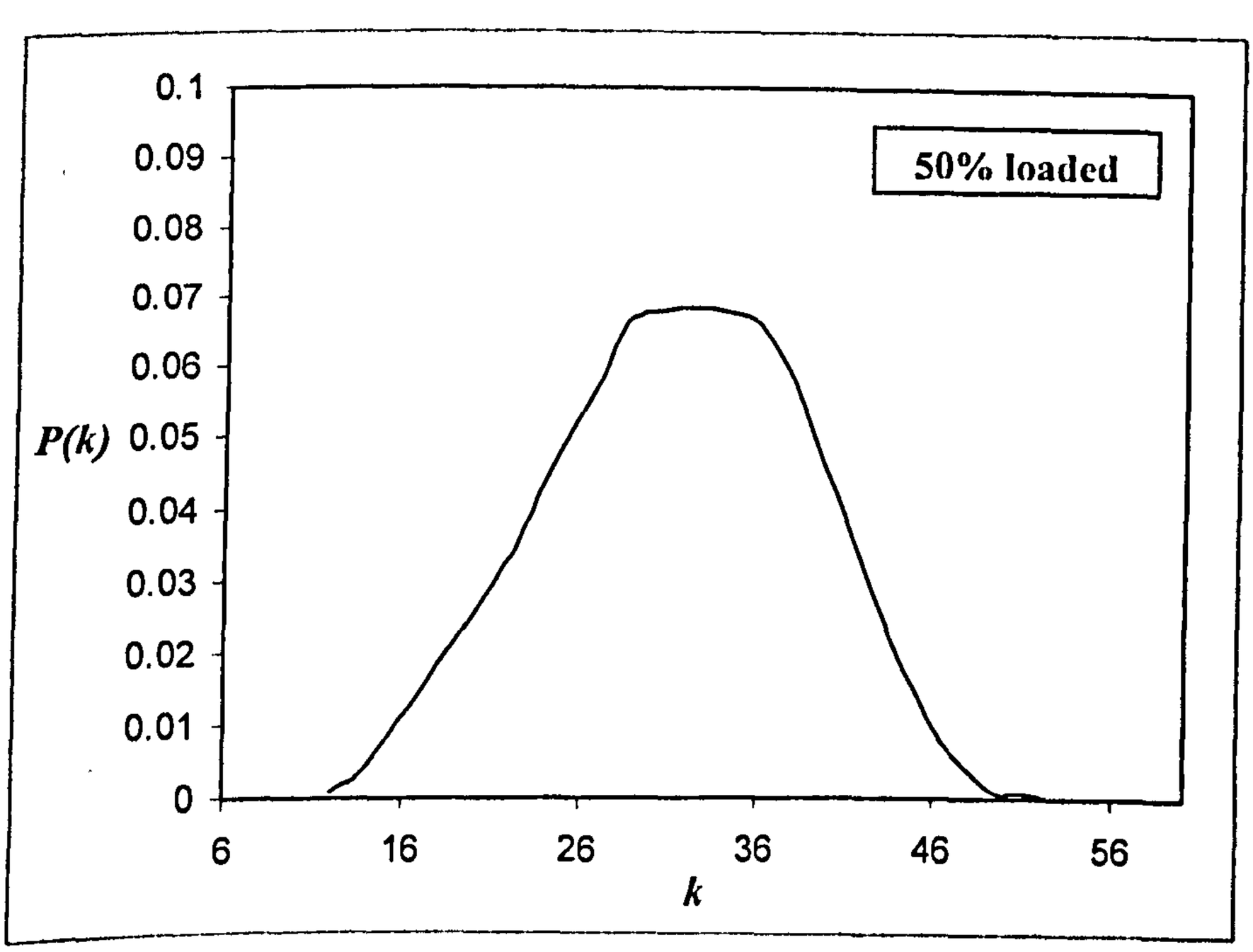


Figure 6.4 The in-degree distributions when the network is loaded up to 50% of it capacity with *N=1024* and *M= 64*.

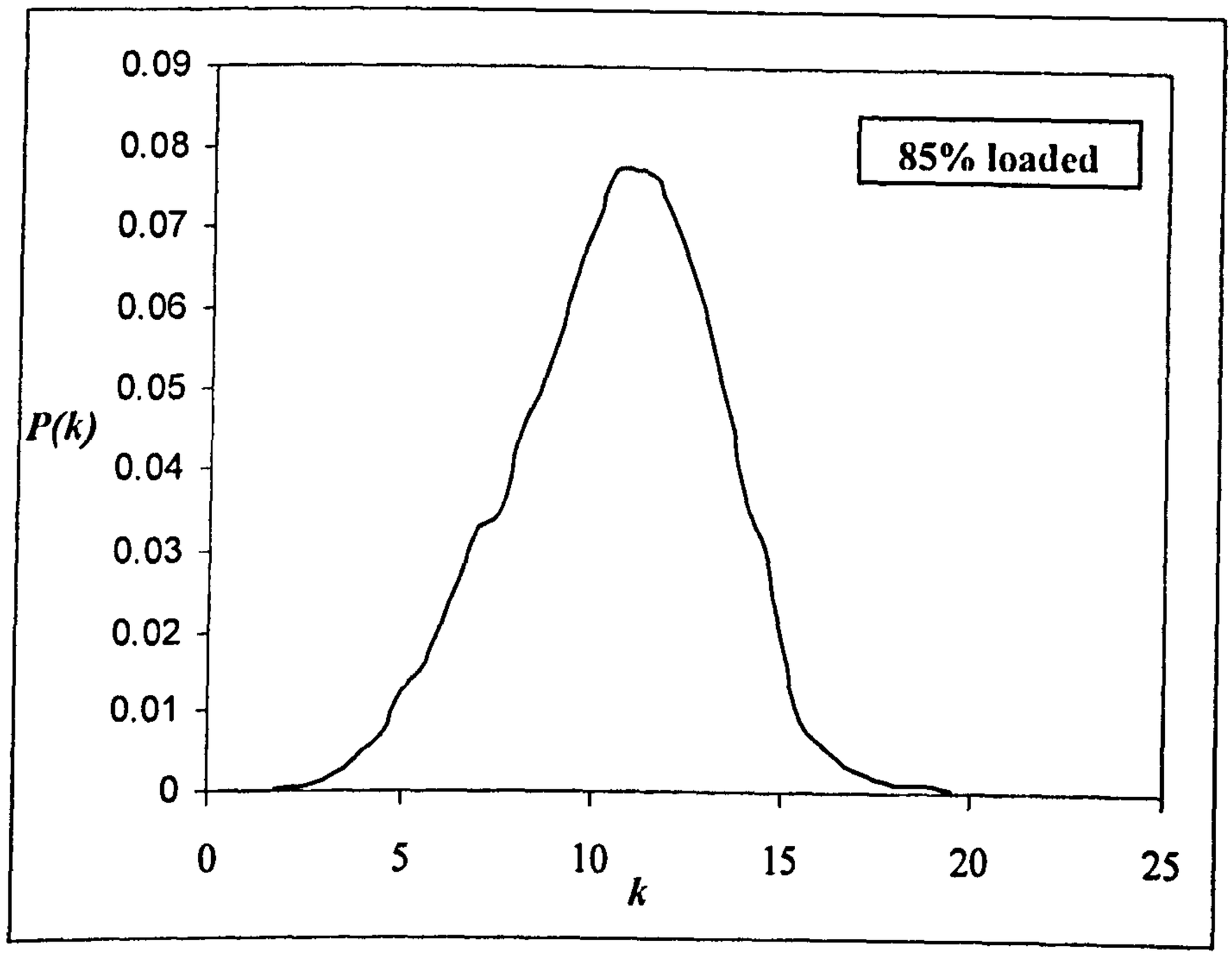


Figure 6.5 The in-degree distributions when the network is loaded up to 85% of it capacity with $N=1024$ and $M=64$.

6.4 Scalability Performance of the RS Load-Balancing Scheme

Simulation results depicted in Figures 6.6 to Figure 6.8 show that the proposed algorithm is scalable and that the generation of regular graphs using biased random sampling is effective for various network sizes. Simulations have been carried out for increasing values of network size and the results presented here demonstrate the true scalability of the algorithm.

As can be observed from theses graphs, the performance of the algorithm scales well specially for large network sizes. The in-degree distributions shown in the following figures are for graphs with network sizes $N=1024$, $N=8192$, and $N=16384$; respectively. In addition, by increasing the number of nodes N, the in-degree distribution is closer to the degree distribution of regular graphs, which indicate that

this algorithm is designed for large-scale networks. Simulations results confirm that the proposed algorithm scales well over large network sizes.

Hence, the generation of ER graphs using the proposed random sampling load balancing algorithm is effective for all network sizes simulated (at least up to $N = 16384$). Moreover, with increasing network size, a convergence to binomial distribution is observed. Consequently, simulations results confirm that the scalability of random sampling (RS) load-balancing scheme over large network sizes.

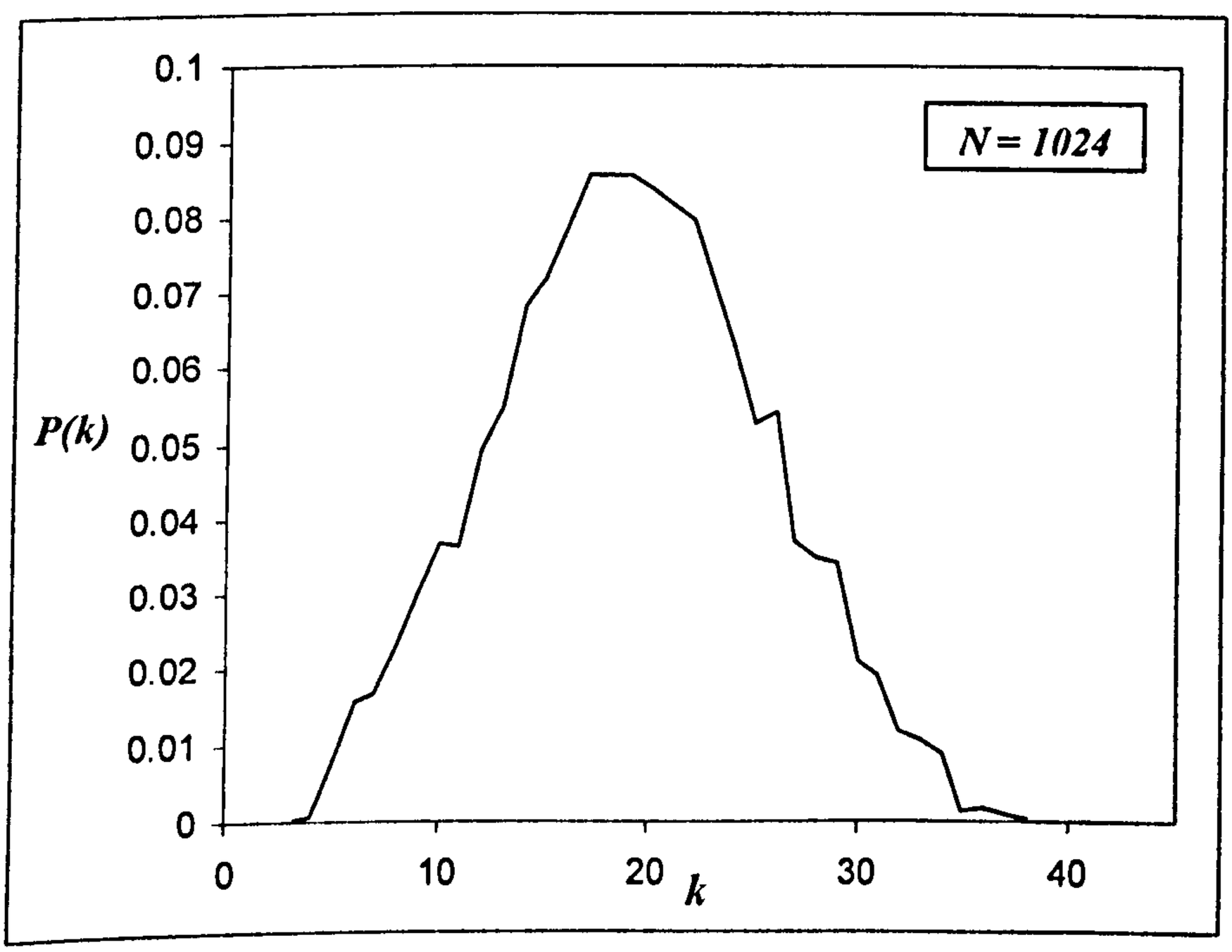


Figure 6.6 The in-degree distributions for a network with $N = 1024$ and $M = 48$. The network is loaded up to 75% of its capacity.

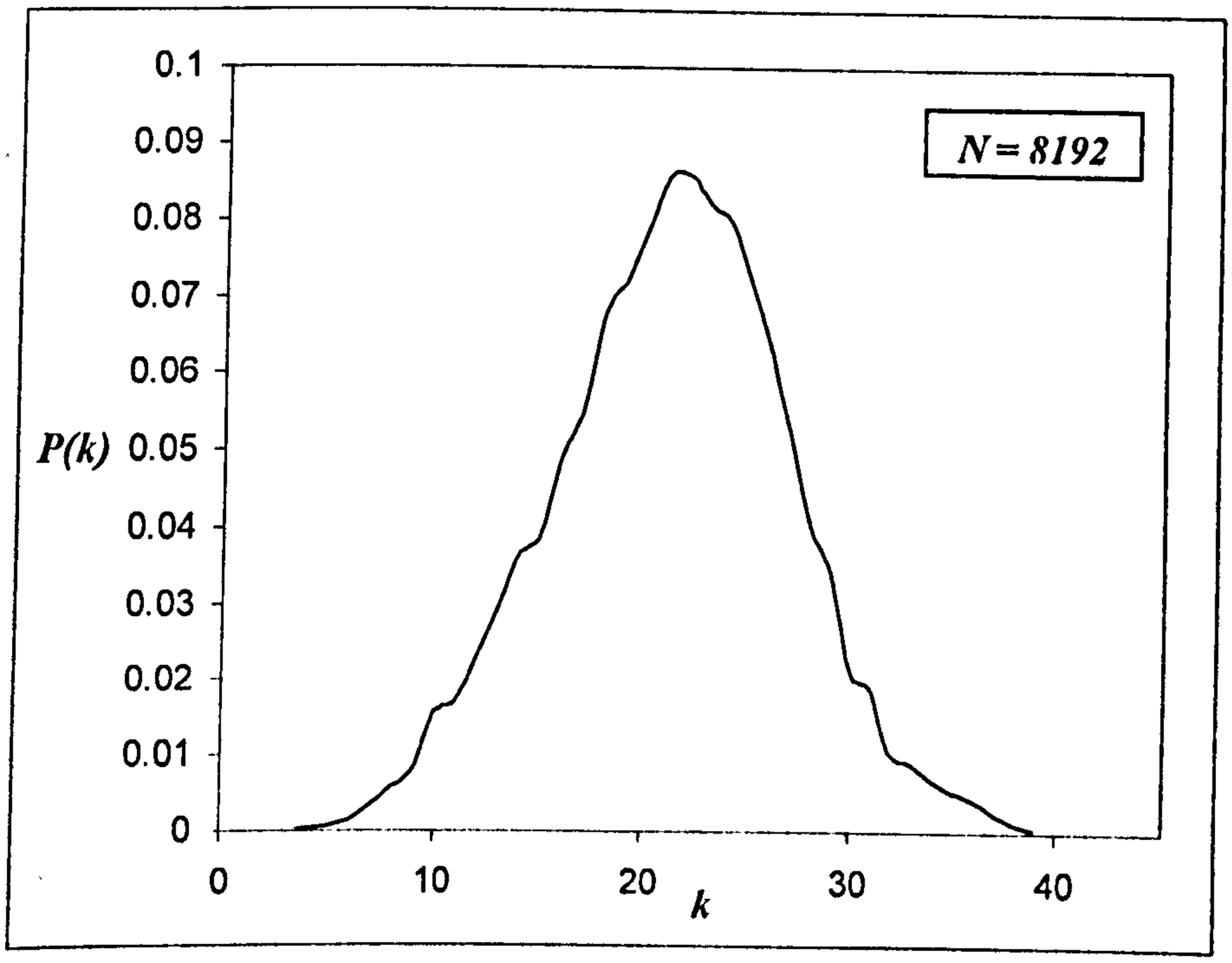


Figure 6.7 The in-degree distributions for a network with $N = 8192$ and $M = 48$.
The network is loaded up to 75% of its capacity

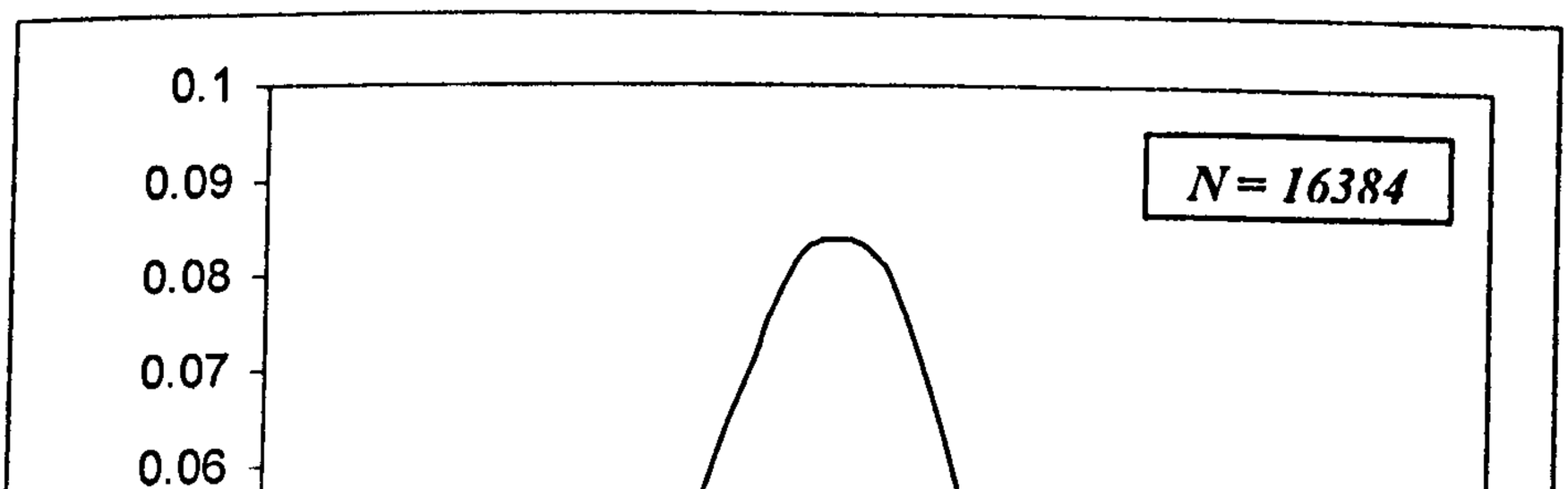


Figure 6.8 The in-degree distributions for a network with $N = 16384$ and $M = 48$.
The network is loaded up to 75% of its capacity

6.5 Performance of the improved BRS load-balancing scheme

The performance of the load-balancing technique proposed in Chapter 5 can be improved by biasing the sampling toward specific nodes instead of choosing them randomly. Hence, the nodes' selection will be based on a predefined criterion (such as computing power, communication latency, and geographic position) rather than choosing the last node in the walk. For example, the random sampling walk would be directed towards unvisited nodes or it would prefer a node with certain properties to allocate the new job.

Accordingly, the BRS load-balancing technique has been improved by assigning the new job to the least loaded (or highest in-degree) node in the sampling, instead of the last node in the walk. When the node with the most-free resources on a random sampling is preferred to receive the new job, its resources must be greater than or equal to the resource of the last node on the random sampling. Therefore, it is expected that the BRS scheme will have the same scalability as the standard random sampling, and the balancing performance is much improved.

ER random graphs have a binomial distribution that exponentially decreases which provides good load-balancing. However, since every node in the network should have the same in-degree, the optimal degree distribution would be a regular graph as the nodes would have the same load. Therefore, improving the proposed load balancing technique to generate more regular graph can enhance the load-balancing performance. Thus, the improved load balancing technique implies that to have an optimal balanced network, the network should be close to a regular graph where all the nodes in the graph have the same in-degree.

It is known that regular graphs have zero standard deviation and zero variance since all the nodes in the graph have the same degree (Cvetković, et al., 1998). The variance is the average of the square of the distance of each possible value from the expected value, and it is used to capture the degree that the data point is being spread out from average value. Hence, if the network system has an expected (average) in-degree $\bar{k}$ edges, then the variance $\mathrm{Var}(k)$ of the network is given by (Loeve, 1977):

$$Var(k) = \sum_k p_k (k - \bar{k})^2 \qquad (6.2)$$

However, the graph can also have a zero standard deviation if it has an even number of nodes. Another balanced network is a network where half of its nodes have the expected in-degree $\bar{k}$, and the other half have in-degree $(\bar{k}+1)$ or $(\bar{k}-1)$. In this case, the in-degree standard deviation is $+0.5$ and -0.5 respectively (i.e. the variance is equal to 0.25). Accordingly, the network is also considered a balanced network when its variance is close to 0.25.

Though it has not been proved that the modified load balancing scheme would generate a regular graph, simulation results for the BRS algorithm demonstrate the same property as a regular graph. Therefore, this scheme can be considered as an improvement of the random sampling technique derived in Chapter 5 which proved analytically to produce ER random graphs. Accordingly, it is expected that the improved load balancing scheme would generate networks with in-degree variances smaller than or equal to the in-degree variance of ER random graphs. The improved load balancing scheme which selects the node with highest degree in the sampling

should balance the load at least the same as the load balancing scheme which generates ER graphs through selecting the last node in the walk for the assigning the jobs.

The proposed biased random sampling (BRS) technique will be used to balance the load distribution and the edge insertion and deletion strategy assures that the load will be distributed equally across all the nodes in the network. Moreover, to prove that the BRS scheme truly samples the nodes preferentially to their in-degree and that the load-balancing performance is much improved, the following simulation results are provided.

6.5.1 Performance evaluation for the BRS scheme

The set of Figures 6.9a to 6.9c demonstrate the in-degree distribution of the network plotted as the network evolved through different time slots (T), which shows the process of reaching the load balancing. Here the time dynamics of the in-degree distributions of the network can be clearly seen. In Figure 6.9(a), the network is initialised in a completely random state. For instance, in this experiment, the network started with a variance approximately equal to 46.3. Then with time, the network starts reshaping itself by balancing the load distribution among the nodes, and in-degree variance decreases to become approximately 11.4 at $T = 2500$; as can be seen Figure 6.9(b). Over time, the network settles down to a nearly regular graph with variance approximately 0.32 at $T = 4000$ as seen in Figure 6.9(c).

Hence, the BRS scheme attained an efficient load-balancing performance regardless the initial status of network's in-degree distribution. By using the BRS protocol, the

network would dynamically reshape itself till it settles down to become nearly regular graph. The time it will take to settle down and balance the jobs depends on network size and its in-degree distribution status (more details are discussed in Section 6.6).

Thus, when all the nodes have the same capabilities, the network will almost be a regular graph and an excellent load-balancing performance is achieved. Thus, the improved BRS load-balancing scheme which selects the node with highest degree in the walk improved the load-balancing performance attained from the previous version which generates ER graphs through selecting the last node in the walk.

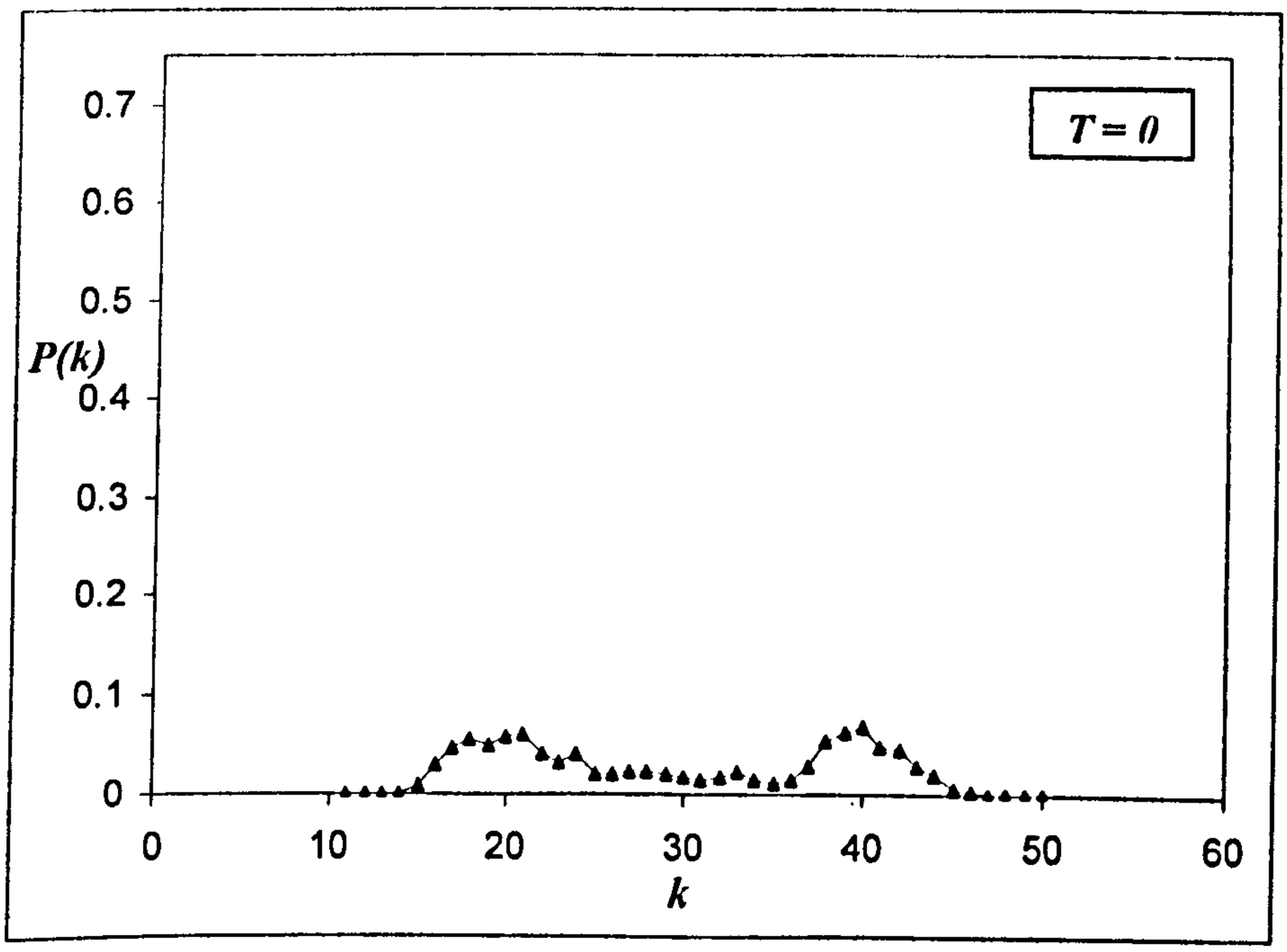


Figure 6.9(a) The in-degree distribution of the network plotted at the time slot $T = 0$.

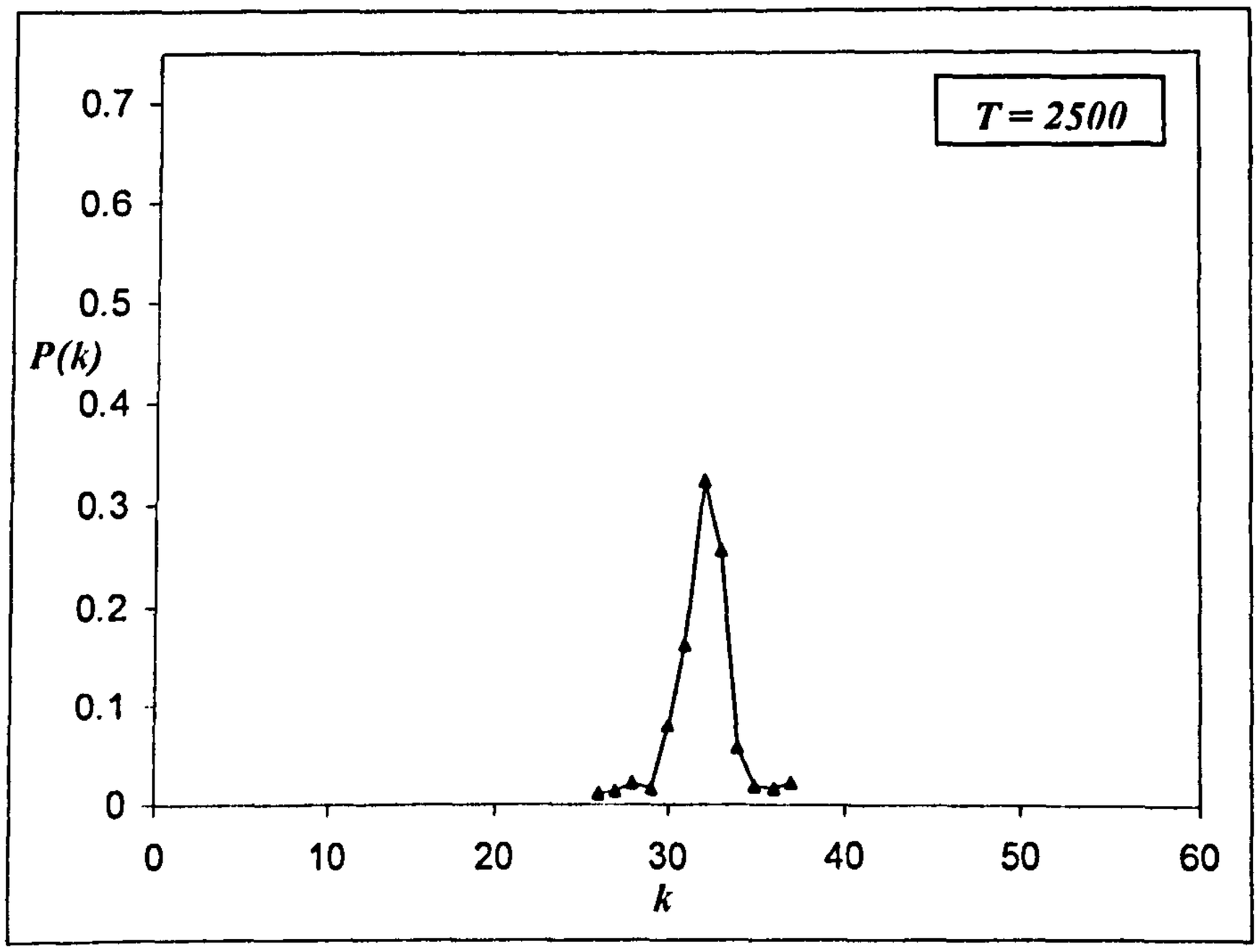


Figure 6.9(b) The in-degree distribution of the network plotted at the time slot $T = 2500$.

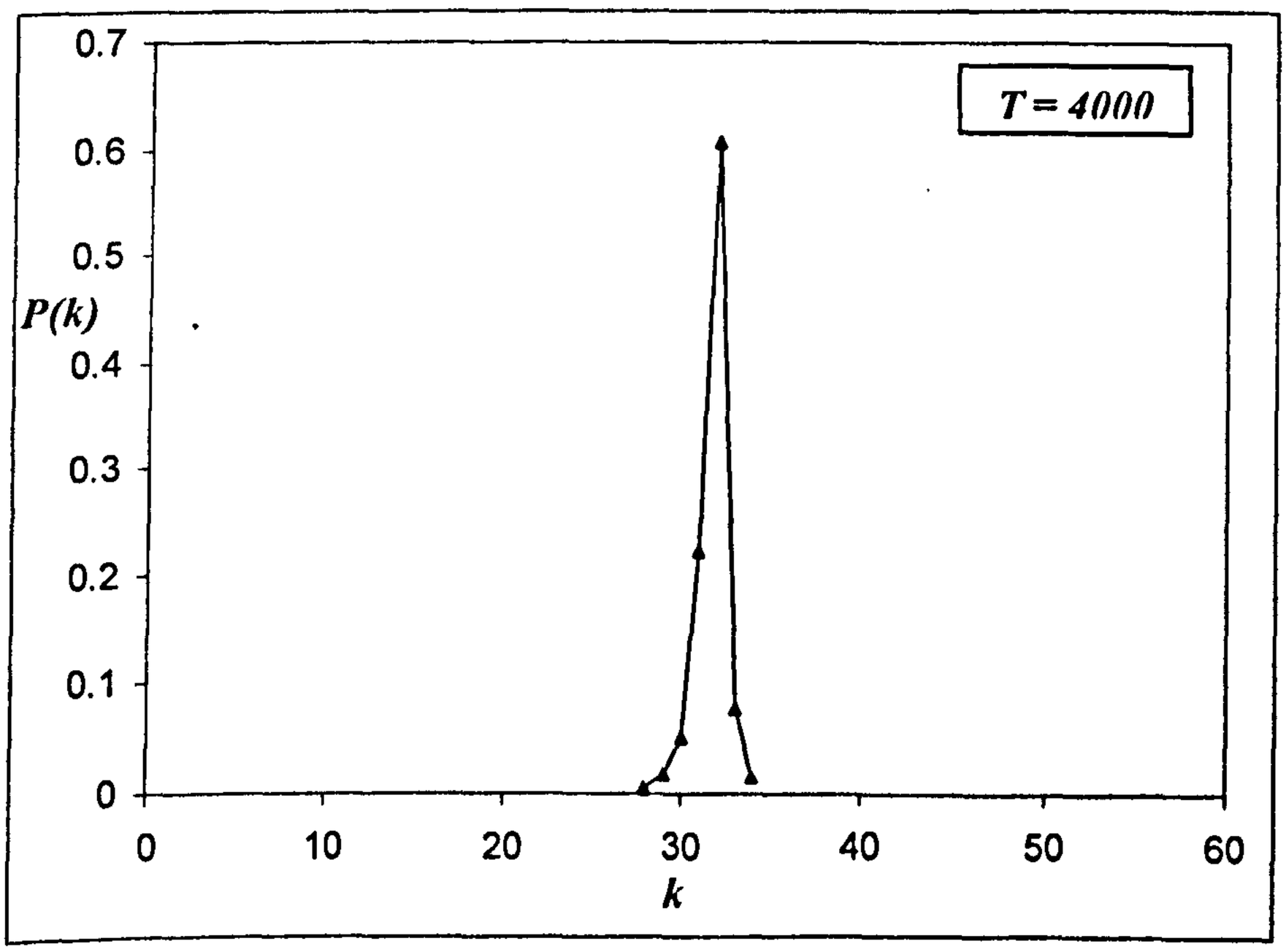


Figure 6.9(c) The in-degree distribution of the network plotted at time slot $T = 4000$.

Figure 6.9 The in-degree distribution plotted as the network evolves over a number of time slots (T).

6.5.2 The justification for using the BRS scheme to balance workload

In order to determine the relation between each node's visitation frequency and its in-degree, the number of times that each node is visited after the network has evolved and balanced is recorded. The random sampling protocol shows that there is a relation between a node's degree and the frequency at which a random sampling visits a node. A linear relation between a node's in-degree and visitation frequency has been observed which validates the use of BRS scheme as a decentralised load balancing algorithm. I also observed that each node is visited at least once which determines the lower limit of the diameter characteristic of the network.

Figure 6.10 shows the variance of the in-degree distributions of the network with time as the network evolves. The network is initialised randomly; for example, the network begins with in-degree variance of approximately 42.6. Then, the network starts reshaping and randomises itself with time by adding and deleting nodes' edges to reach a large in-degree variance of around 63.3. Then, the network starts to settle down and begins to heal itself. Consequently, the variance rapidly decreases until the network becomes almost regular with an in-degree variance close to 0.38.

Hence, the BRS load-balancing protocol will efficiently and dynamically randomise and reshape the network to reach a nearly regular graph nevertheless the in-degree status of the network. The time required to heal the network and balance the workload depends on network size and its in-degree variance status. But, the time required by the network to settle down is significantly dependent on network size as discussed in Section 6.6.

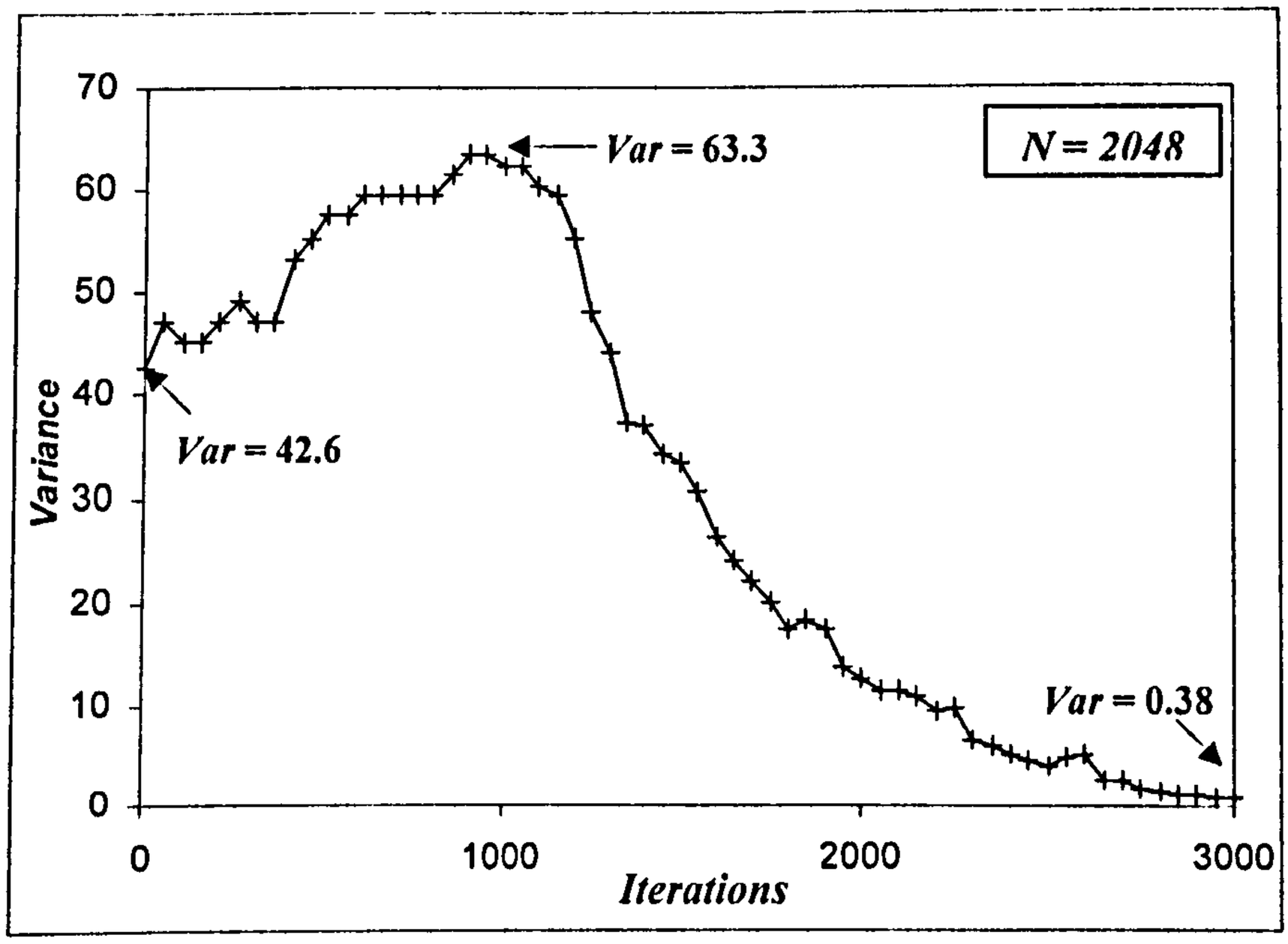


Figure 6.10 The variance of the in-degree distribution vs. Time for a network with $N = 2048$ and $M = 48$.

6.6 The Time required to Balance the Network in BRS Scheme

Now, to further measure the efficiency of the BRS algorithm, the average time required for the network to settle down and balance the load is examined. Simulations have been carried out for networks with different sizes and the average time it took by each network to balance the load distribution is recorded. Table 6.1 shows the network size and the corresponding aver time required to reach variance around 0.32, which is close to the optimum variance to perfectly balanced network. In Figure 6.11, the network size is plotted with the corresponding time required for balancing the network. As seen from the figure, the time required to efficiently balancing the network load increases logarithmically with network size (N).

The above simulations have been carried out not only to illustrate relationship between the load-balancing time and network size, but also to capture the dynamics

of network growth. Theses simulation results confirm that the proposed BRS technique is suitable for growing networks.

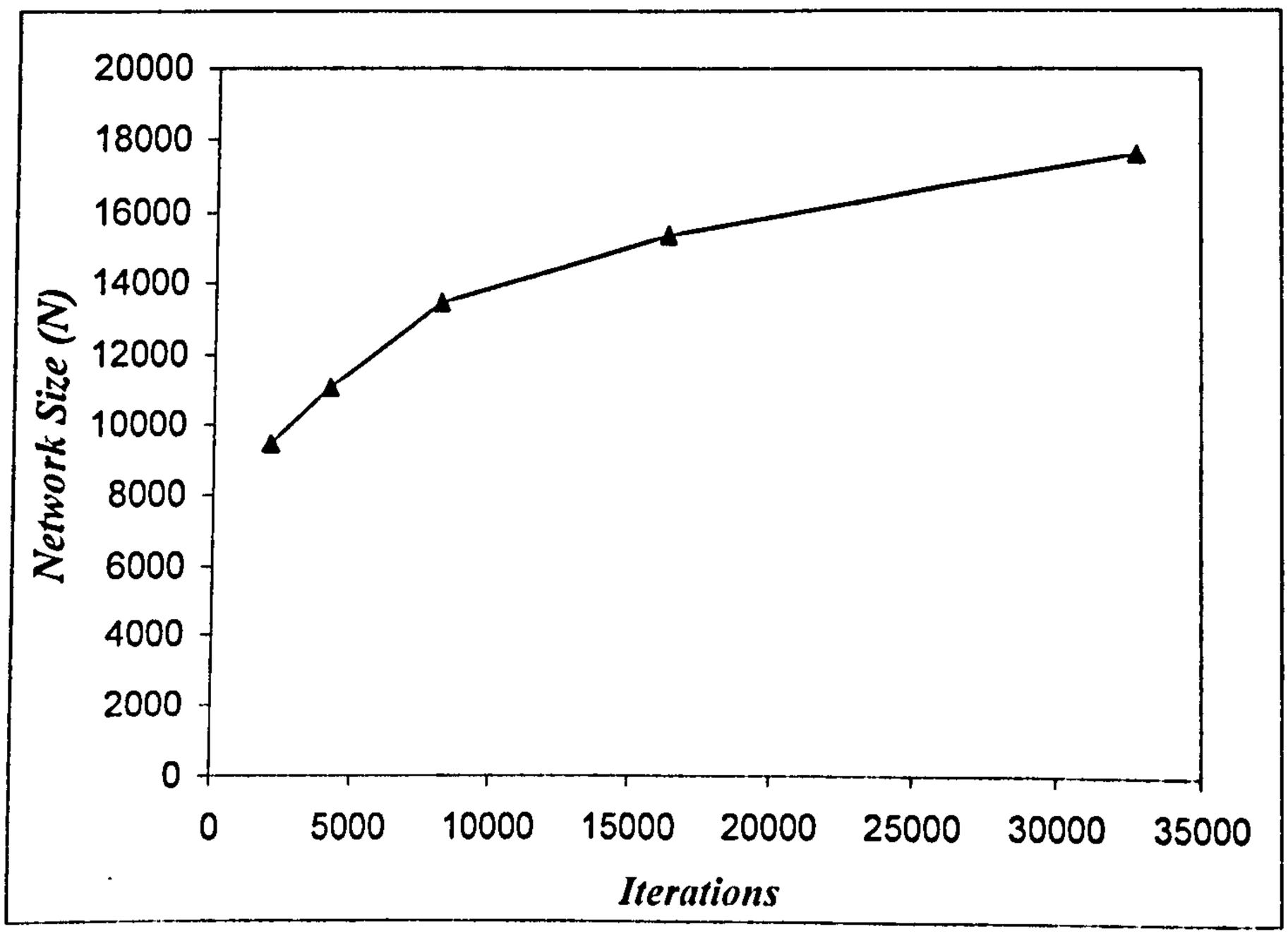


Figure 6.11 The time required to balance the network increases logarithmically with the network size N.

Table 6.1 The time required to balance the network for different network sizes

Network Size	Average Time to Balance (Iterations)
512	5188
1024	7244
2048	9444
4096	11111
8192	13518
16384	15370
32768	17592

6.7 The Optimum Random Sampling Length

Intensive simulations have been carried out to observe the number of steps needed to efficiently sample the network to achieve the required load distribution, and to evaluate its effect on the performance of load balancing algorithm. I found that the performance of the load-balancing algorithm improves as the random sampling length increases.

As can be seen from Figure 6.12, increasing the random sampling length will decrease the in-degree variance. Here, simulations were performed on a network of 2048 nodes with several random sampling lengths. If the random walk sampling is too short, then the load distribution is not very efficient and the variance is very high. This is because the network builds an overloaded cluster near the node that is initiating the job (and thus the random sampling). This cluster stays overloaded while the rest of the network is unloaded. However, if the random sampling length is 16 or more (for a network with 2048 nodes), the in-degree variance is small and very close to 0.25, which is the optimum variance for balanced networks.

Moreover, if the number of steps used to sample the network is very large, then the decrement in the in-degree variance is very small. As shown in Figure 6.12, the performance achieved by using very long sampling walk is very close to that when a random sampling of length close to $log(N)$ is used. This has also been observed in larger network sizes. Intensive Simulations have been carried out on different network sizes to examine the number of steps needed to efficiently sample the network, and the results are recorded in Table 6.2.

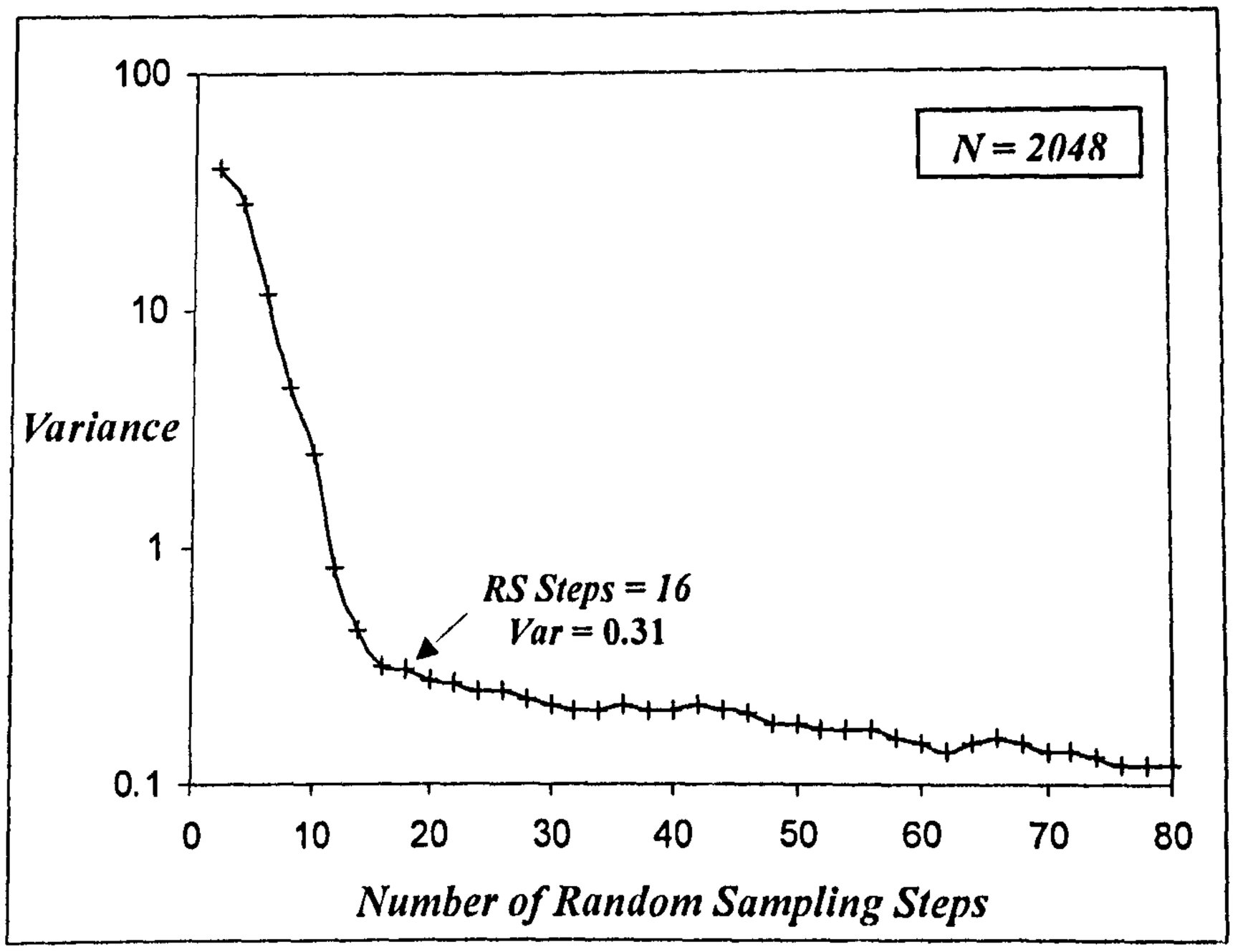


Figure 6.12 The variance of the in-degree distribution vs. number of random sampling steps for a network with $N = 2048$ and $M = 48$.

It can be observed form the table that using random sampling with length around $log(N)$, as a cut off, will be sufficient to reach an in-degree variance very close to the optimum variance, and this confirms that the random sampling technique is very efficient to balance the workload in the network.

Table 6.2 The random sampling length and the variance resulted in different network sizes.

Network Size	Number of Random Sampling Steps	Variance
512	9	0.3
1024	13	0.29
2048	16	0.31
4096	17	0.33
8192	19	0.34

6.8 Reliability Performance of the BRS Scheme

To measure the reliability and robustness of the proposed load balancing mechanism, the simulations are extended to investigate how nodes' in-degree and load distribution in the network will be affected by random errors at run time. To do this, the possibility of node failure is introduced to the network after the network has settled down and distributed the load properly among the nodes. Node failure can occur due to node errors or attack. Here, it is assumed that the number of nodes that will fail is a random variable with Poisson distribution.

Figure 6.13 shows the in-degree variance with time during random attack and node errors. It can be observed from the figure that the variance is increased dramatically when nodes failed. However, the network starts to heal itself and dynamically reshapes itself by re-distributing the load between the nodes. As a result, the in-degree variance will rapidly decrease and the network will become almost regular again. Thus, the proposed load-balancing scheme is reliable and robust against random errors or attack. Thus this algorithm enables the network to dynamically and efficiently reorganise and heal it-self against node failure or attack.

Simulation results prove that the proposed BRS scheme does not require a specific connection to be available to randomly sample the network. If a particular edge is missing due to failure, another node or path can be used without affecting the load-balancing performance. Nevertheless, maintaining a reliable communication between the nodes is very important as these edges are used to maintain network status and losing data packets will prevent jobs from being transferred and executed.

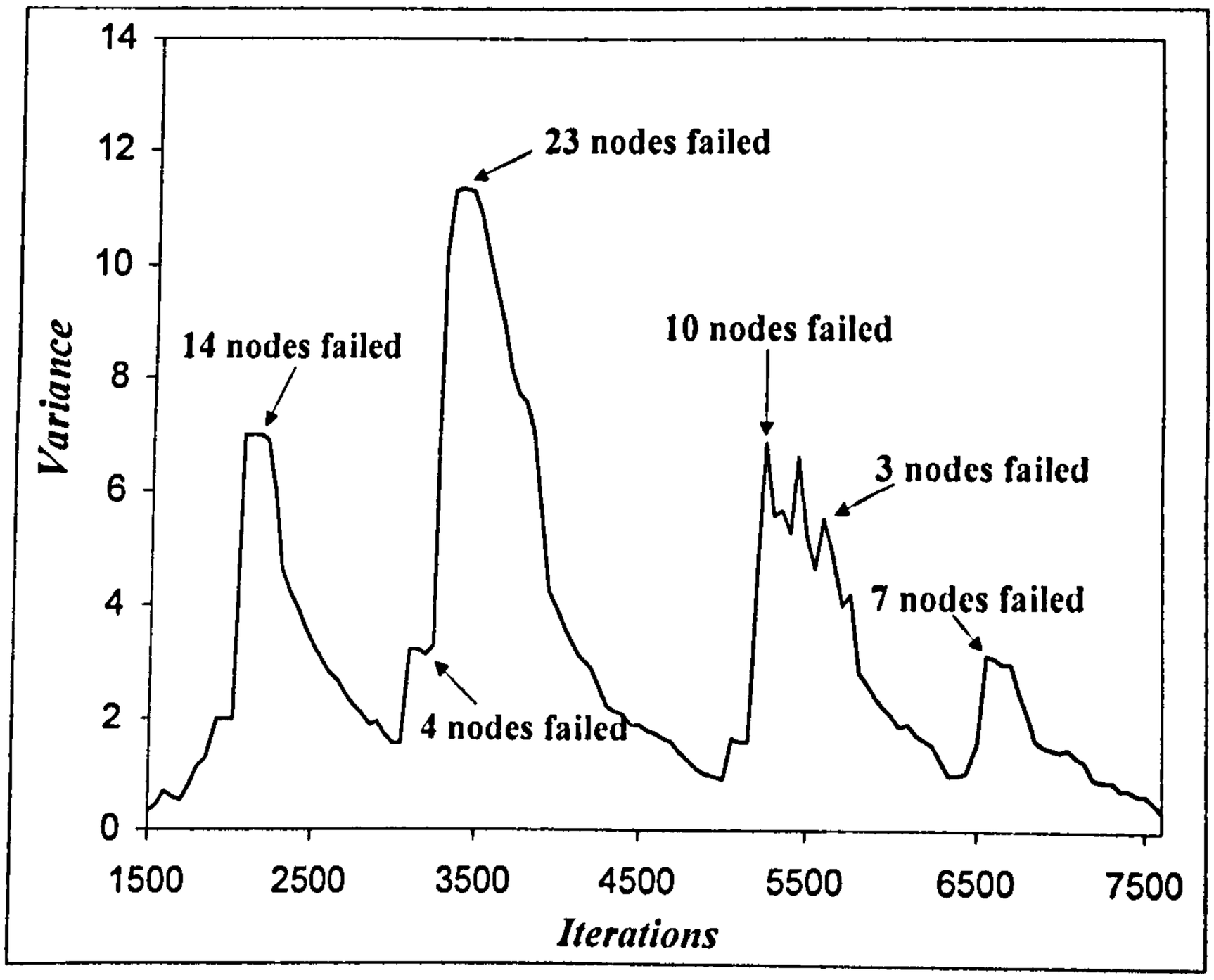


Figure 6.13 The in-degree variance for a network affected by random attack.

$N = 2048$ and $M = 48$.

Therefore, designing and implementing efficient protocols to achieve a reliable transmission between the nodes while minimising latency will play a key rule for BRS scheme implementation.

6.9 Latency-Optimised Load Distribution Algorithm

In many network systems, load balancing is not restricted to the use of resources or computing power alone, but also is influenced by the geographical distance and communication delay between the nodes. Therefore, locality information has been included into the random sampling scheme to improve the proposed load-balancing technique for such network systems. The latency-optimised load balancing scheme models the network as a graph consisting of biased nodes' power and communication edges. Therefore, the load balancing scheme goal is to balance the

load between the nodes as well as to minimise the communication latency. Accordingly, the random sampling will prefer a geographically closer node or that has lower latency even if it is not the highest degree on the walk. This has been implemented by adding the geographical distance and communication delay factors in the random sampling process while distributing the load. The round trip delay time (RTT) data measured for a Grid Networks from the PingER Project (PingER, 2005) has been used to model the communication delay in simulations. The RTT values are between $30msec$ and $160msec$ (PingER, 2005).

Two experiments carried out and the average round trip latencies (RTL) for each executed job has been reported. As expected, simulation results prove that adding the locality factor in the load distribution algorithm indeed reduced the overall network latency. Table 6.3 summarises the simulation results for the average latencies observed in both load-balancing schemes; the latency-optimised scheme and the latency in the original scheme for different network sizes. As we can see from the table, the overall network latency is reduced up to 22% by using latency-optimised load-balancing scheme.

Table 6.3 The latencies observed in both the latency-optimised scheme and the latency in the original scheme for different network sizes.

Average Latency (msec)	*N=512*	*N=1024*	*N=2048*
LB Scheme	90.28	96.47	99.72
Latency-Optimised LB Scheme	74.06	76.73	77.81
Improvement	17.9%	20.5%	21.9%

Figure 6.14 shows the latency for completed jobs recorded for both load-balancing schemes. The network under consideration consists of 512 nodes distributed within a radius of 1500 kilometres. As can be observed from the figure, in the original load-balancing scheme, the overall average latency for the completed jobs is $90.28m\,\text{sec}$, whereas in the latency-optimised scheme, the overall average latency for the completed jobs is $74.06m\,\text{sec}$. Thus, the overall average latency has decreased from $90.28m\,\text{sec}$ to $74.06m\,\text{sec}$. That is, the latencies maintained by using the latency-optimised scheme have been reduced by at least 18% on average compared to those observed in the non-geographic aware scheme. The most important observation noticed here is that the latencies for the individual completed loads when using the latency-optimised algorithm will always remain close to the average latency without big overshoots (fluctuations), which make the network stable and reliable and a suitable environment for applications that require specific quality of service.

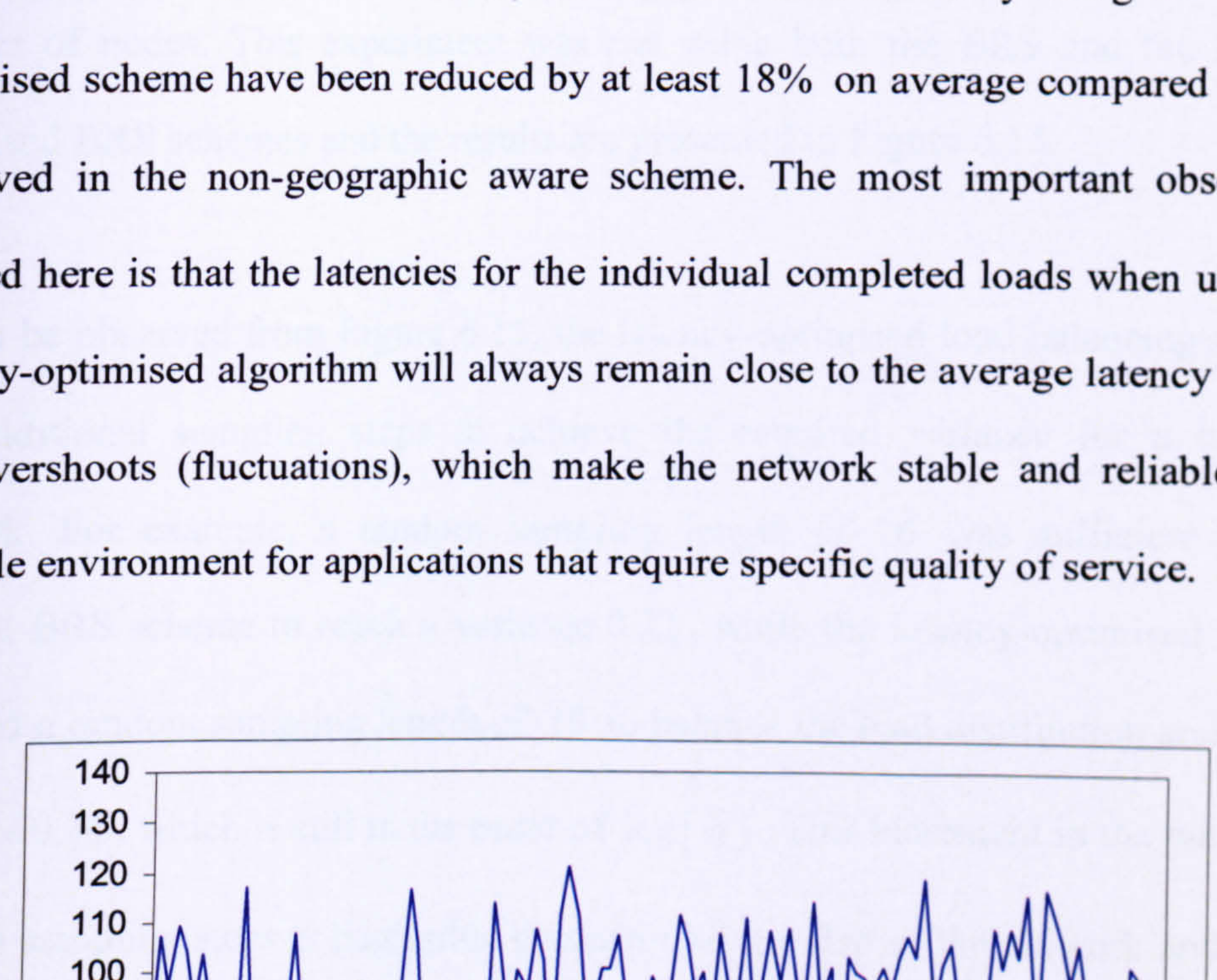


Figure 6.14 The average round trip latencies observed for the finished jobs in a network with $N = 512$.

Now, to study the effect of adding the locality factor on the load-balancing performance, the number of sampling steps required to efficiently balance the network under consideration is studied and compared with the original scheme. In this experiment, a network with 2048 nodes is considered, and the random sampling length required efficiently distributing the load and reaching an in-degree variance close to the optimal one has been measured. I have also arrived at a generic expression that links the cut off value for the random sampling length with the number of nodes. This experiment was run using both the BRS and the latency-optimised BRS schemes and the results are presented in Figure 6.15.

As can be observed from Figure 6.15, the latency-optimised load balancing requires few additional sampling steps to achieve the required variance for a balanced network. For example, a random sampling length of 16 was sufficient for the original BRS scheme to reach a variance 0.32 , while the latency-optimised scheme required a random sampling length of 20 to balance the load distribution and have a variance 0.34 , which is still in the order of $\log(N)$. This increment in the number of random sampling steps is negligible compared to the size of the network and would be acceptable given that the improvement in latency achieved.

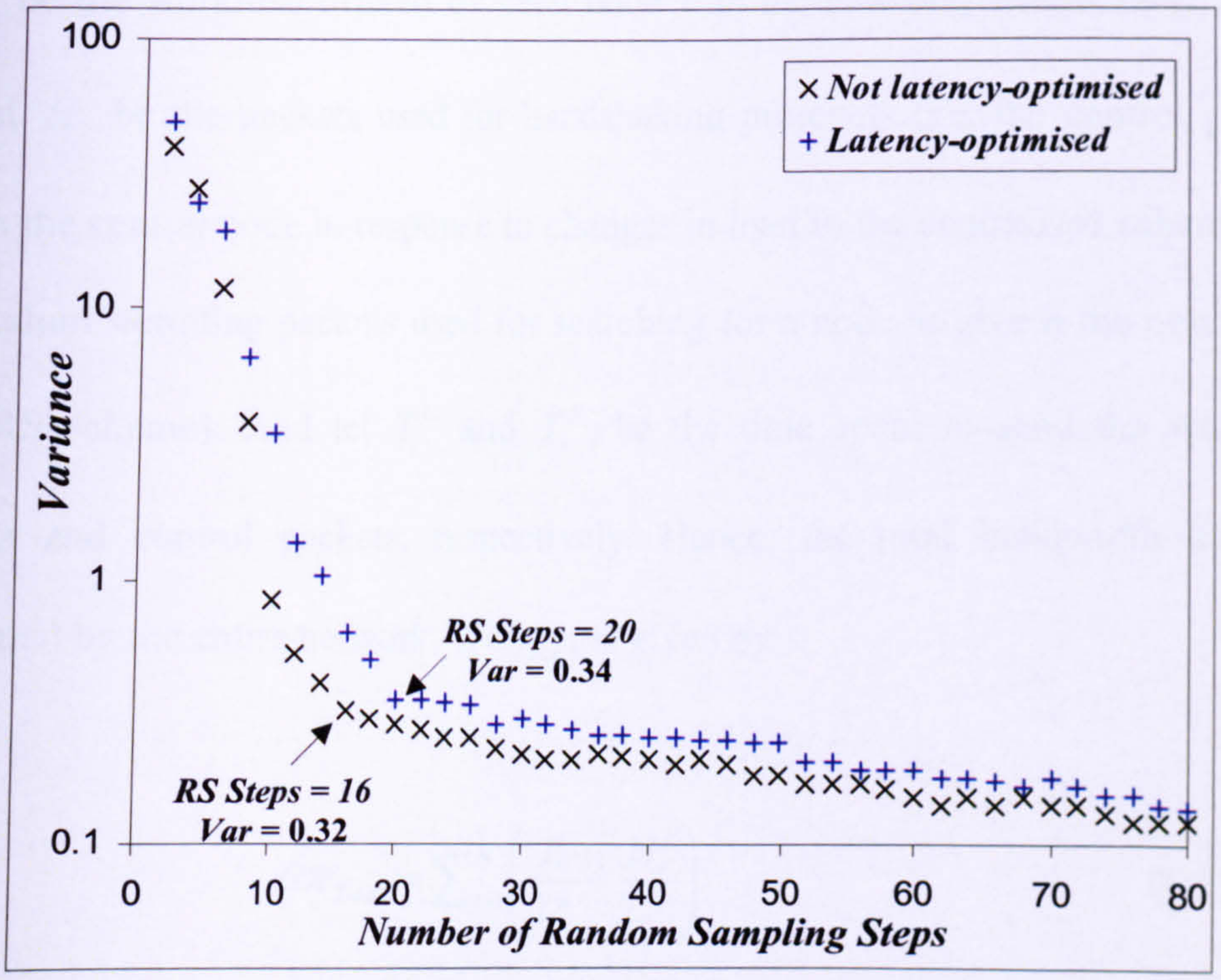


Figure 6.15 Comparison of the variance vs. random sampling (RS) length for a network with $N = 2048$.

6.10 Performance Comparison of the BRS Scheme with the Centralised Algorithm

To evaluate the performance of the proposed biased random sampling algorithm, two important performance measurements in distributed systems were examined: the total job throughput achieved and the bandwidth capacity required by the load distribution mechanism. Then, the performance of the biased random sampling scheme is compared with the performance of the centralised mechanism.

6.10.1 Bandwidth Capacity Analysis for the BRS and the Centralised Algorithms

In this section, the bandwidth capacity occupied by the BRS algorithm is analysed and compared with the centralised algorithm. For a network system with N nodes,

let L_i be the workload offered by each node i in the network, where $i \in \{1,...,N\}$, and let H_i be the packets used for handshaking protocols (i.e. the control packets sent to the central node in response to changes in load in the centralised scheme, and the random sampling packets used for searching for a node to give it the new job in the BRS scheme). And let T_i^L and T_i^H be the time spent to send the workload packets and control packets, respectively. Hence, the total bandwidth capacity consumed by the entire network, BW_{Total}, is given by

$$BW_{Total} = \sum_{i=1}^{N} \left(\frac{L_i}{T_i^L} + \frac{H_i}{T_i^H} \right) \tag{6.3}$$

In the centralised scheme, the central node has to know the load status in each of the nodes that are in the network. Therefore, the central node needs to periodically check the status of every node in the network, and the nodes have to inform the central node if they finished executing the jobs so that the central node can update the network load status. Therefore, for a network with N nodes, the central node will have to check the status of all the N nodes in the network before distributing the load. As a result, the total bandwidth capacity consumed by the network is in the order of network size N; that is the order of $O(N)$.

For the BRS algorithm, each node that initiates a new job must initiate a random sampling to search for a node to assign the new job. And since the random walk will be $O(\log N)$ length, the total bandwidth capacity of the random sampling will be in the order of $O(\log N)$. Therefore, for a network of N nodes, the total bandwidth

capacity occupied by the biased random sampling algorithm will be in the order of $O(N \log N)$, which is greater than the total bandwidth capacity required by the centralised scheme .

In Figure 6.16, the total bandwidth capacity consumed by the network for both algorithms recorded for different network sizes. As can be observed from the figure, the central server algorithm indeed requires less total bandwidth capacity than the BRS algorithm.

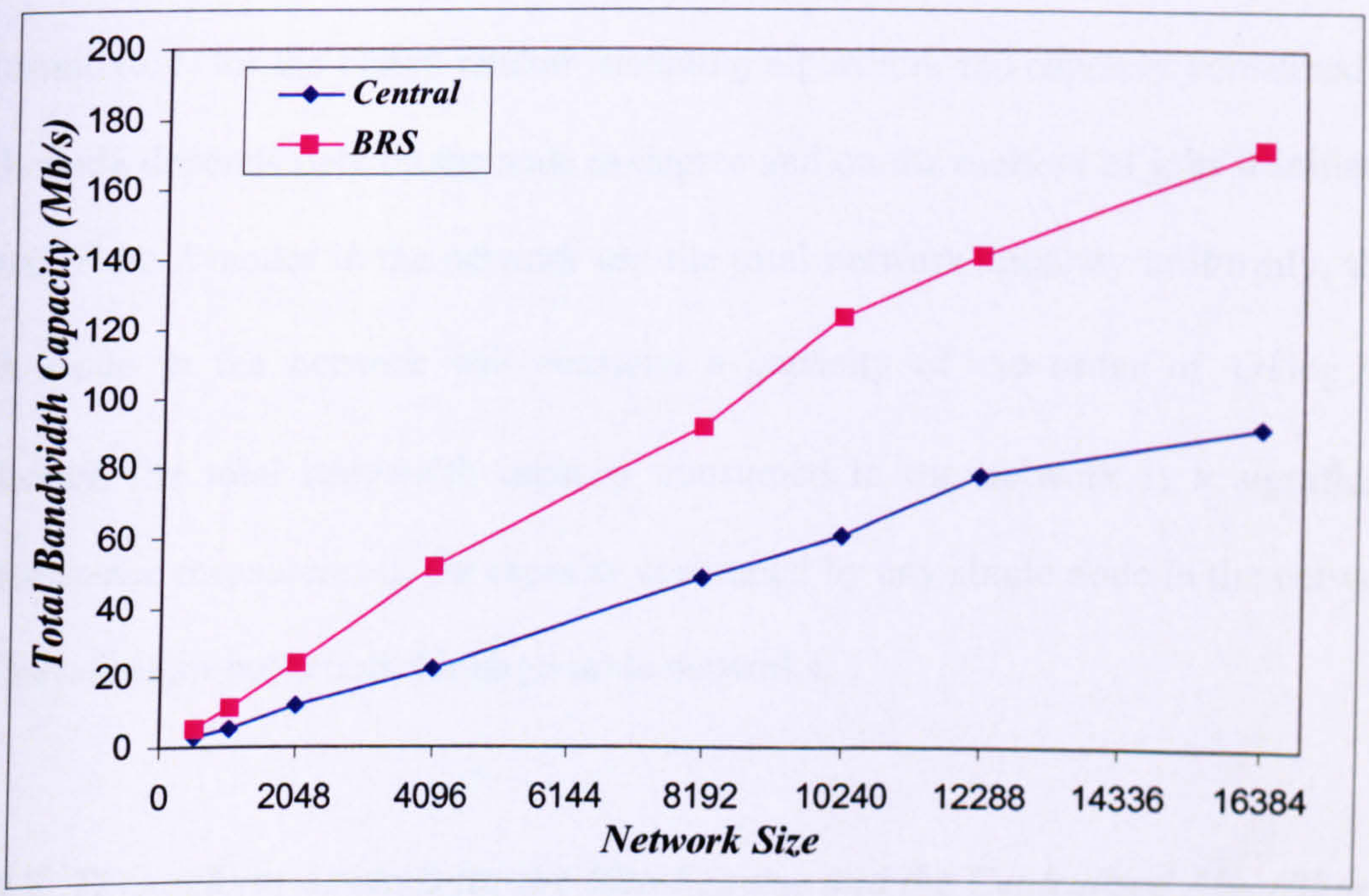


Figure 6.16 The total bandwidth capacity consumed in the network recorded for different network sizes.

Another important performance factor to consider here is the bandwidth capacity consumed by individual nodes in the network. The bandwidth capacity required by each node in the network, BW_i, is given by

$$BW_i = \frac{L_i}{T_i^L} + \frac{H_i}{T_i^H} \quad , \quad i \in \{1,...,N\} \tag{6.4}$$

A very interesting observation was made here for biased random sampling scheme is found to considerably lower the bandwidth capacity consumed by any node in the network when compared to that for the centralised scheme and moreover stays highly stable for any network size as shown in Figure 6.17. Since the central node in the centralised scheme is engaged in all jobs and handshaking transfers, the central node consumes a network capacity of $O(N)$.

Alternatively, for the biased random sampling algorithm, the capacity consumed by each node depends only on the node in-degree and on the number of jobs it initiates. Thus, if the N nodes in the network use the total network capacity uniformly, then each node in the network will consume a capacity of the order of $O(\log N)$. Although the total bandwidth capacity consumed in the network is a significant performance measurement, the capacity consumed by any single node in the network can be of major bottleneck for large-scale networks.

6.10.2 Throughput Analysis for the BRS Scheme and the Centralised Algorithm

Another important performance metric in distributed networks is the system throughput. Throughput is the number of completed jobs during a specified period of time. The objective here is to have the maximum amount of completed jobs (large amount of throughput). Therefore, the total throughput achieved by the biased random sampling algorithm is analysed and compared with the centralised scheme.

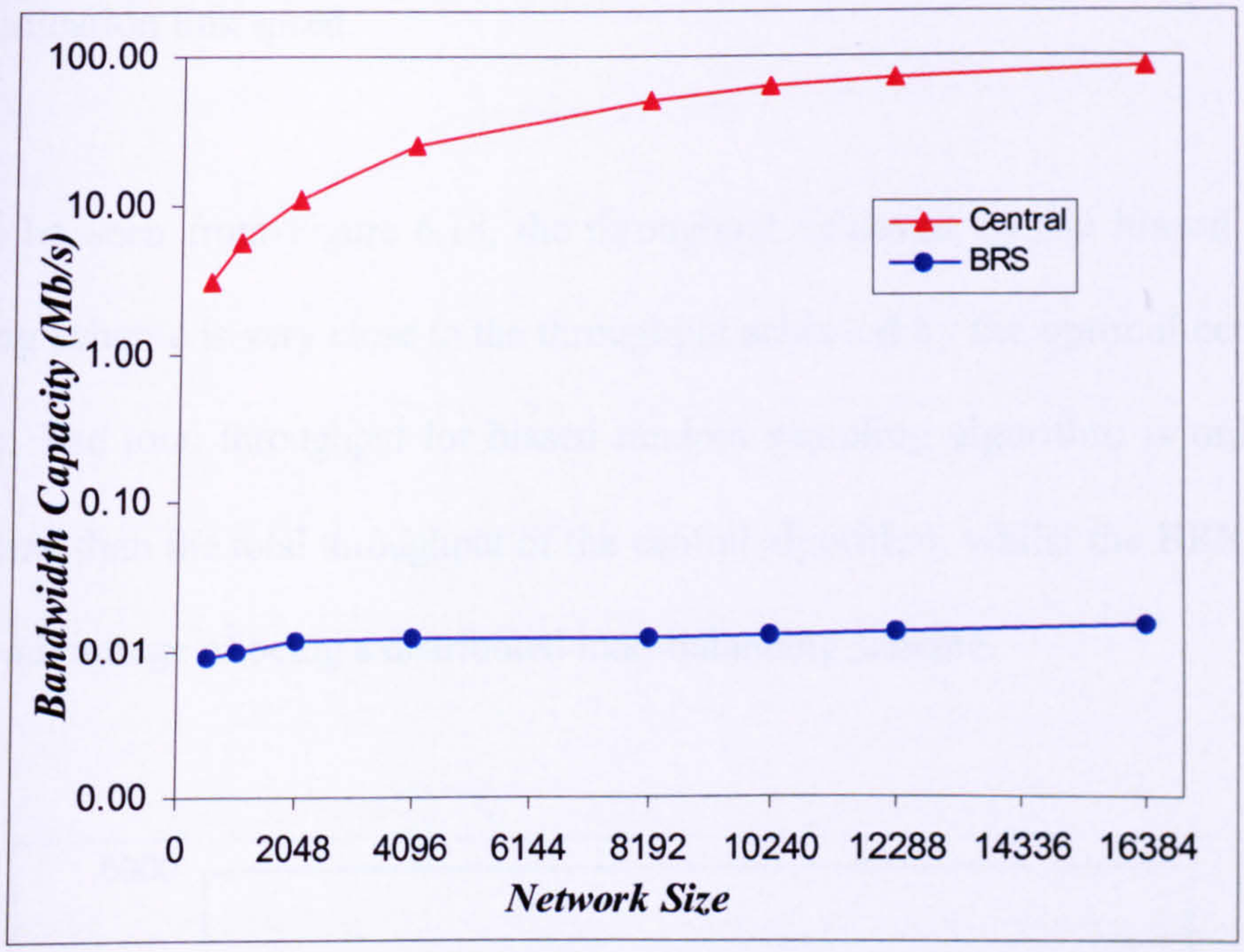


Figure 6.17 The average bandwidth capacity consumed by individual nodes for different network sizes.

For a network system with N nodes, let J_i be the number of completed jobs for node i in the network where $i \in \{1,...,N\}$. Let $T_{Simulation}$ be the total simulation time. Hence, the total throughput attained in the network, $Throughput_{Total}$, is given by

$$Throughput_{Total} = \frac{\sum_{i=1}^{N} J_i}{Time_{simulation}} \tag{6.5}$$

Figure 6.18 shows simulation results for the total throughput achieved by both the central load balancing scheme and the biased random sampling scheme. In these simulation results, it is assumed that the nodes in a network have equal capabilities, and the job sizes and arrival rates follow Poisson distribution. Moreover, I took into account the effect of communication delay on the total throughput performance by distributing the nodes in a network of 1500 kilometres radius area with $10Mbps$

communication link speed.

As can be seen from Figure 6.18, the throughput achieved by the biased random sampling scheme is very close to the throughput achieved by the optimal centralised scheme. The total throughput for biased random sampling algorithm is only about 3% worse than the total throughput of the central algorithm, whilst the BRS scheme has the advantage of being a distributed load-balancing scheme.

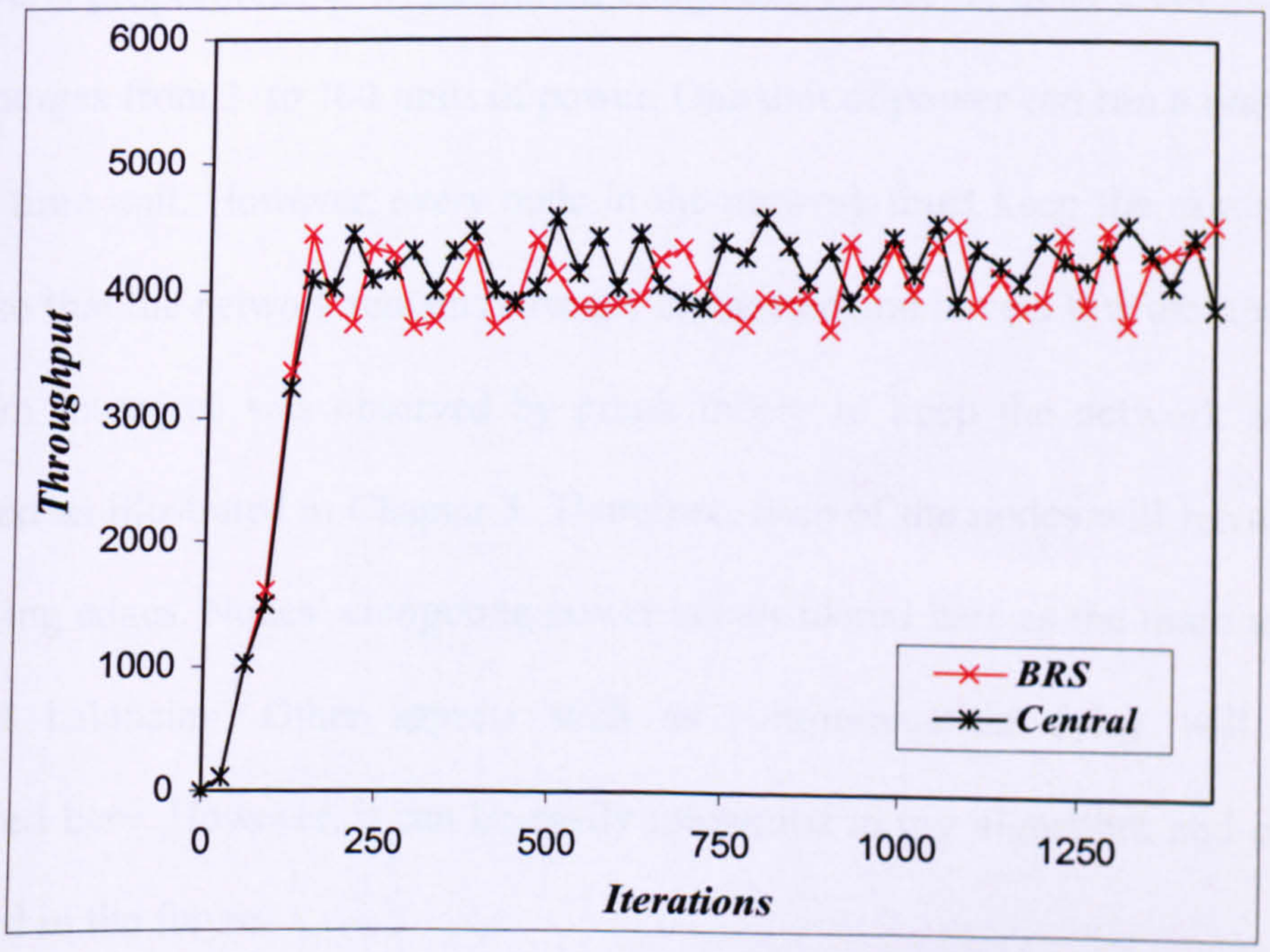


Figure 6.18 Simulation results for network throughput achieved by both the central load-balancing scheme and the biased random sampling scheme.

6.11 The Performance of the BRS Scheme in a Heterogeneous Network Systems

A number of potential improvements to my load-balancing technique and generalisations of my model deserve further study. Here, to further measure the

efficiency of the proposed biased random sampling (BRS) mechanism for load balancing under more strict conditions, simulations has been extended to include heterogeneous nodes. In such systems, the nodes will have different capabilities and varying resources. This will help us in understanding how these situations will affect the nodes' in-degree distribution and the load distribution in the network.

For heterogeneous system simulations, each node in the network has a number of edges equal to its computational power (capability). The maximum in-degree a node can have is proportional to its maximum computing power. A node's computational power ranges from 1 to 100 units of power. One unit of power can run a unit of load in each time unit. However, every node in the network must keep the minimum in-degree so that the network remains strongly connected and have a low diameter. This minimum in-degree was observed by graph theory to keep the network strongly-connected as illustrated in Chapter 5. Therefore, each of the nodes will have at least 4 incoming edges. Nodes' computing power is considered here as the main aspect of the load balancing. Other aspects such as communication delay will not be considered here. However, it can be easily integrated in my algorithm and could be addressed in the future.

As with homogenous systems, the in-degrees of the nodes are kept proportional to their free resources. And since the random sampling will select nodes preferentially to the in-degree of the nodes, it will preferentially sample the nodes according to their free resources. And due to the correlation between the load and the in-degree, the random sampling will select the node with the highest in-degree in the walk.

For simulations purposes, Pareto distribution has been used to model nodes' computation power distribution and job arrival rate. The degree distribution is given by the following equation (Barabási, 2002, Mitzenmacher, 2003):

$$P(k) = k^{-1}, \quad 4 \le k \le 100 \tag{6.6}$$

The jobs arrival rate is given by Equation (6.7):

$$P(\nu) = \nu^{-1}, \quad 1 \le \nu \le \nu_{max} \tag{6.7}$$

Here, ν_{max} is the maximum number of jobs that can arrive at any time in the network, and for heterogeneous system experiments, it is set to 4096 jobs at each time step.

The Pareto distribution is one type of Power-law probability distributions. Power-law distribution is commonly used to represent several real network systems (Mitzenmacher, 2003). Therefore, Pareto distribution is also used to model job sizes, and in heterogeneous system simulations, it is assumed that size of jobs ranges from 32 Kbits to 2048 Kbits. This is an important distribution since in most real networks they have many small jobs and few larger ones. Therefore, these simulations can give a practical image of how the system will work under real loads.

Since the minimum node in-degree is 4 and the maximum node in-degree is 100, there exist many nodes which will have low power capability. Thus, for highly loaded network, it will be difficult to get close to perfect balancing. Therefore, in

heterogonous systems where many nodes will have low maximum in-degree, the correlation between node's load and its capacity (in-degree) is used to measure its performance.

To determine the relation between each node's computational power and its in-degree, the number of jobs each node received is recorded. In Figure 6.19, the offered load versus node's capacity is plotted for a heterogeneous network system. As we can see from the figure, there is a very close relation between a node's in-degree with the amount of load it receives. This nearly linear relationship between the number of jobs the nodes received with their computational power proves that the BRS load-balancing scheme indeed distributes the load efficiently among the nodes in heterogeneous network system with power-law distribution as discussed.

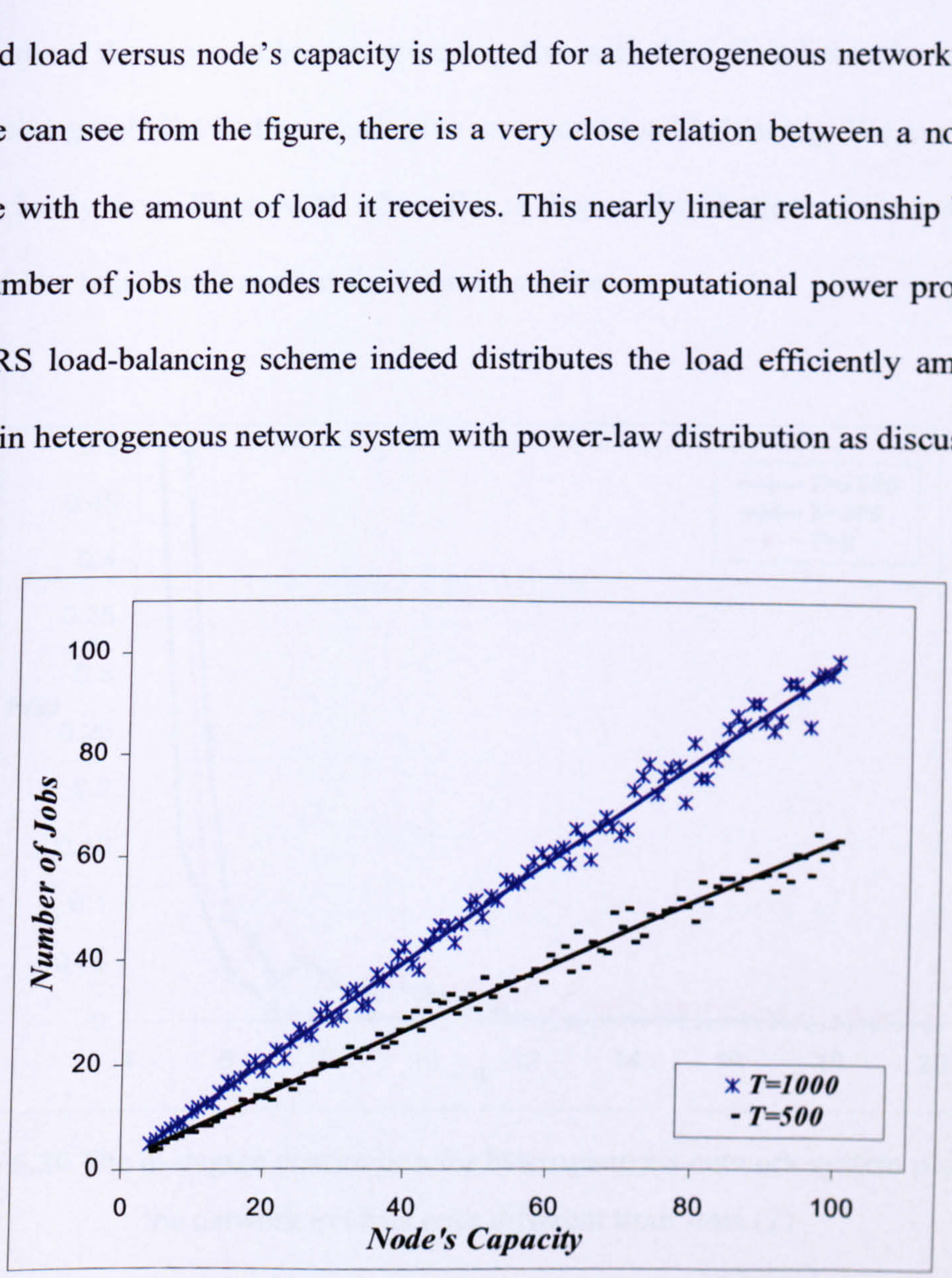


Figure 6.19 The offered load versus node's capacity plotted for a heterogeneous network system.

Figure 6.20 shows the degree distribution for the network with Pareto-law resources distribution in different time slots during load balancing process. As shown in the figure, the degree distribution of the network has a Power-law trend. *Thus, the* degree distribution for the network matches the of the nodes resources distributions for Power-law resource distributions as seen in the figure. Moreover, as the network settles down and the biased random sampling algorithm balances the load distribution, the exponent become more negative with time till it becomes very close to regular graph distribution which give us a good load-balancing. We can see this more clearly from Figure 6.21 where the in-degree distribution of the network is plotted in a logarithmic scale as the network evolves.

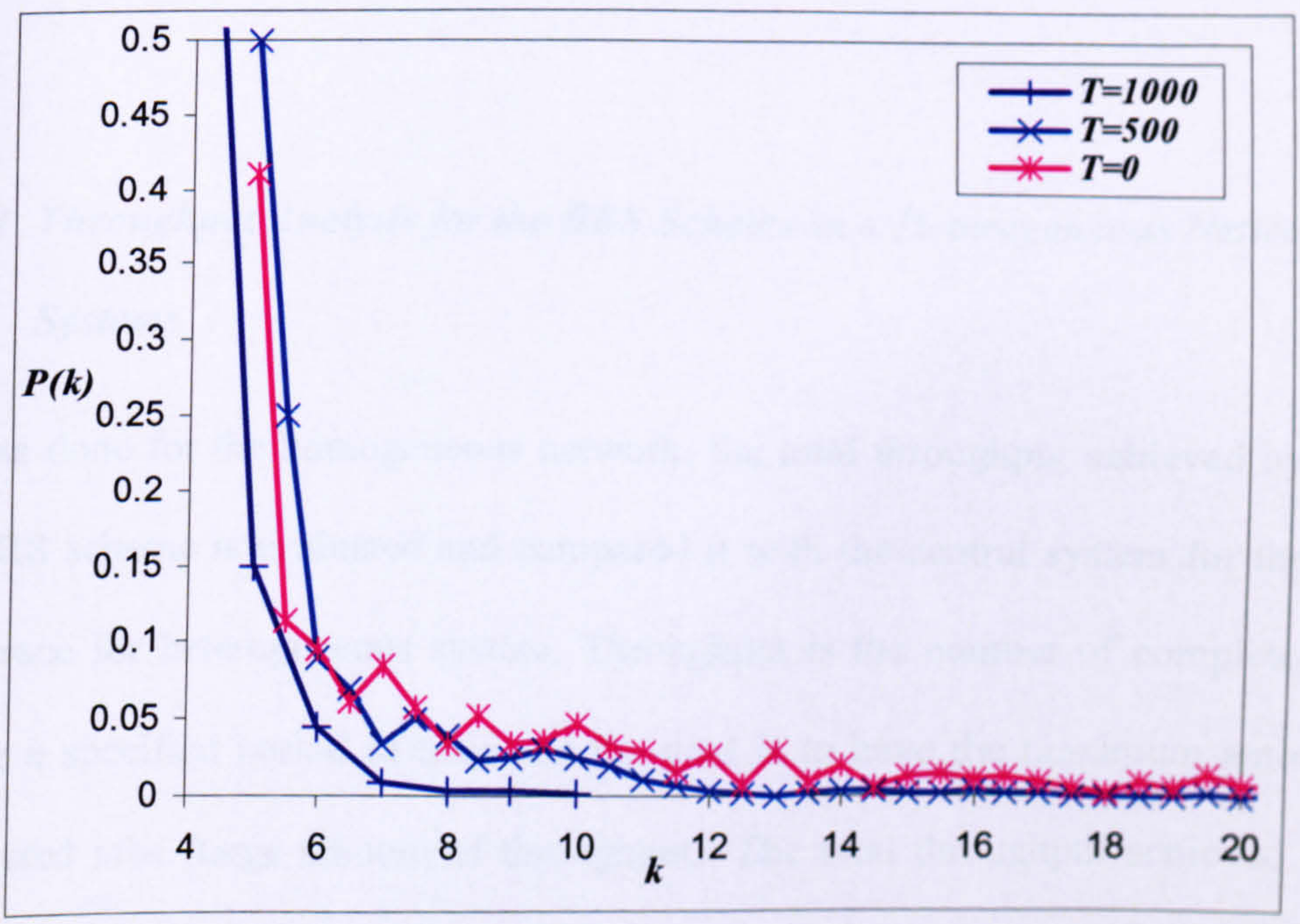


Figure 6.20 The in-degree distribution for heterogeneous network system plotted as the network evolves over different time slots (*T*).

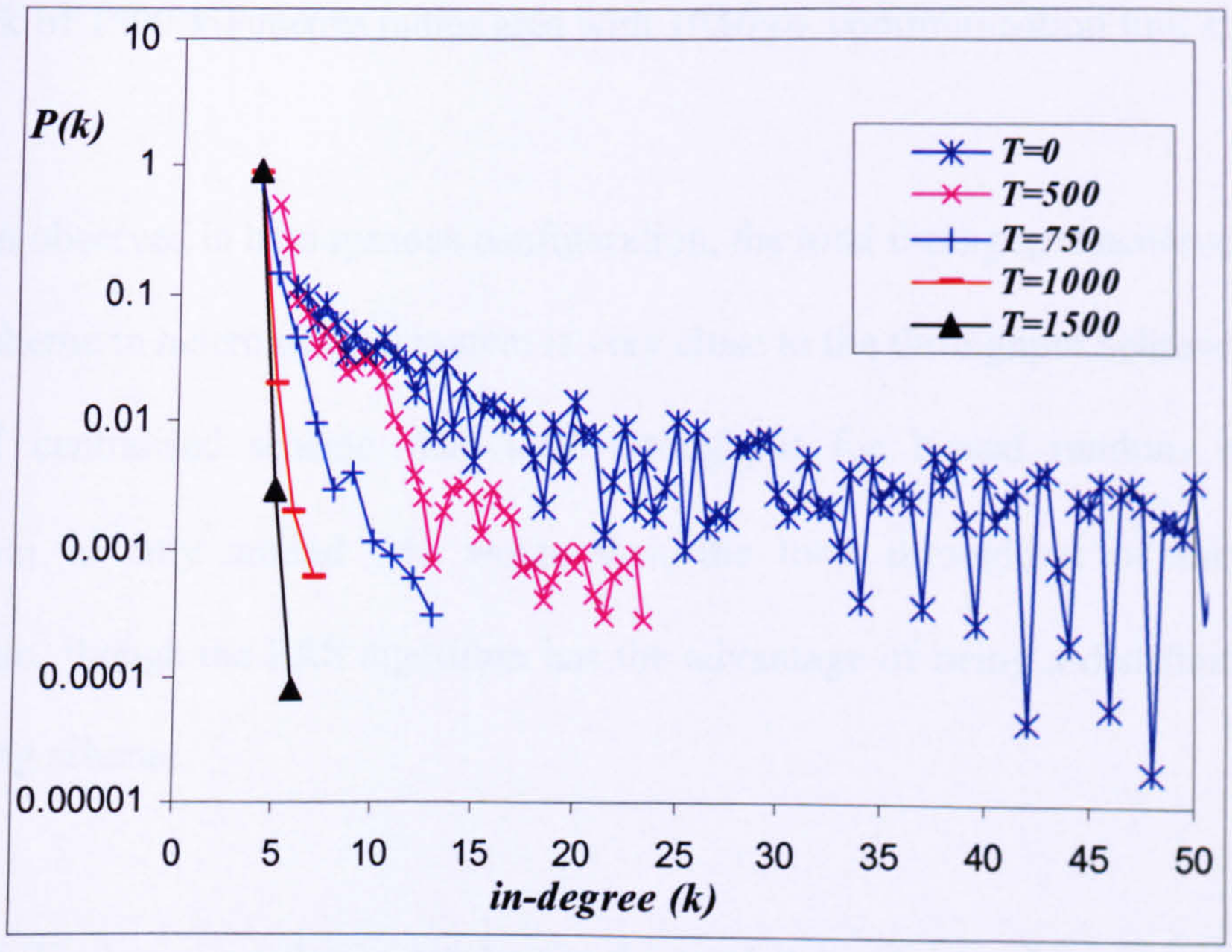


Figure 6.21 The logarithmic plot for the in-degree distribution for heterogeneous network system plotted as the network evolves over different time slots (T).

6.11.1 Throughput Analysis for the BRS Scheme in a Heterogeneous Network Systems

As was done for the homogeneous network, the total throughput achieved by using the BRS scheme is evaluated and compared it with the central system for the same load trace for heterogeneous system. Throughput is the number of completed jobs during a specified period of time and the goal is to have the maximum amount of completed jobs (large amount of throughput). The total throughput achieved in the network is calculated in Equation (6.5).

To properly analyse the total throughput attained by the BRS scheme in heterogeneous system, the effect of communication delay and RTT on the total throughput performance is considered. This is done by distributing the nodes in a

network of 1500 kilometres radius area with 10*Mbps* communication link speed.

As been observed in homogenous configuration, *the total t*hroughput achieved by the BRS scheme in heterogeneous system is very close to the throughput achieved by the optimal centralised scheme. The total throughput for biased random sampling algorithm is only around 3% worse than the total throughput of the central algorithm, though the BRS algorithm has the advantage of being a distributed load-balancing scheme.

Figure 6.22 shows simulation results for the total throughput achieved by both the central load balancing scheme and the BRS scheme in heterogeneous systems.

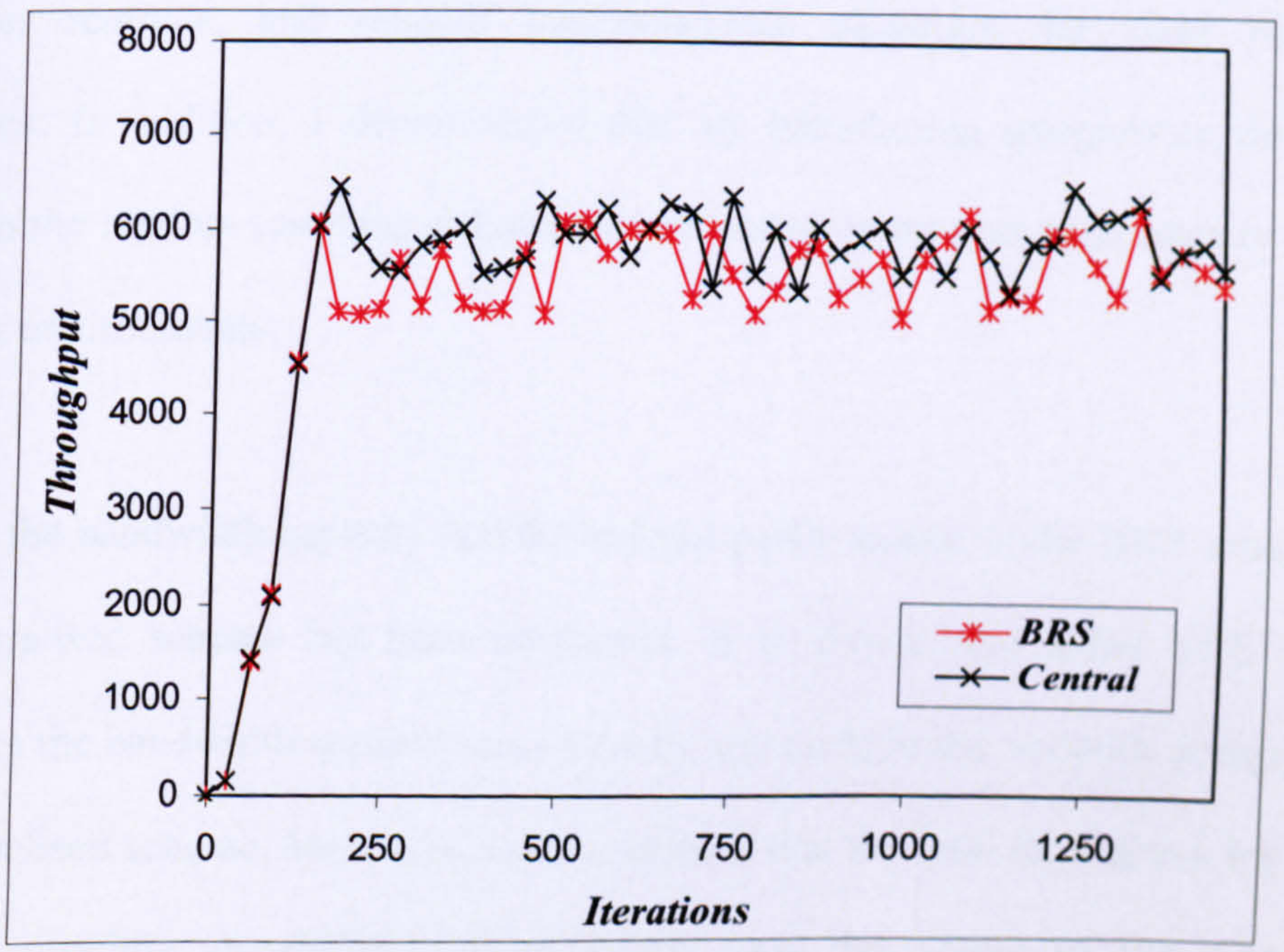


Figure 6.22 Simulation results for throughput achieved by both the central load balancing scheme and the BRS scheme for heterogeneous system.

6.12 Summary

In this Chapter, the performance of the proposed biased random sampling (BRS) load-balancing scheme has been discussed and evaluated. The performance of my load balancing technique under both homogeneous and heterogeneous configurations is examined. The mathematical models and simulated results verify that for homogenous network systems, the proposed network system generates ER random graphs and matches the analytical results.

Then, the proposed load-balancing algorithm has been improved by biasing the random sampling toward the highest in-degree node in the network. Extensive Simulation results show that the generated network system has approximately regular graph degree distribution. Thus, the proposed BRS scheme provides an *effective*, *scalable*, and *reliable* load-balancing paradigm for Grid Networks resources. In addition, I demonstrated that by introducing geographic awareness factor in the random sampling reduces the effects of communication latency in Grid network environments.

Finally, the bandwidth capacity and throughput performance of the BRS scheme and the centralised scheme has been compared. It is shown that using BRS scheme decreases the bandwidth capacity occupied by any node in the network compared by the centralised scheme. Moreover, I demonstrated that the total throughput for biased random sampling algorithm is close to the total throughput of the centralised algorithm, while the BRS scheme has the advantage of being a distributed load-balancing scheme.

The proposed BRS scheme is a straightforward and easily implemental scheme using standard networking protocols. The decentralised feature of the scheme makes it suitable for many applications involving very large number of nodes such as Web Mirroring, PlanetLab (Peterson, el., 2002), and Grid Networks. Moreover, it can be also useful for applications which are designed to handle large parallel computational problems with hundreds or thousands of nodes since it scales well to very large system sizes.

The BRS load-balancing paradigm is a novel technique that can be used for any type of resource sharing and it is not limited to be applied in only distributed computing. The in-degree of a node can be made to correspond to any type of shareable resources. And since the generated networks have low diameters, this makes them easy to sample using random sampling. Simulation results confirm the efficiency of this approach in networks with a large range of resource and load distributions.

Chapter 7

CONCLUSIONS AND FURTHER WORK

Chapter 7

CONCLUSIONS AND FURTHER WORK

7.1 Discussion and Conclusions

In this thesis, an effective, scalable, and reliable load-balancing scheme for the distributed resources accessible on Grid Networks has been demonstrated. A stochastic network system is proposed which provides a distributed load-balancing scheme by generating almost regular networks. The developed load-balancing scheme is based on biased random sampling (BRS) technique to assign new jobs and to update resources availability. Therefore, load balancing is achieved without the need to monitor the nodes for their resources availability.

The main idea of the BRS load-balancing scheme is that the network structure can represent the load distribution status of the network which is dynamically maintained by each node through the links used to connect with other nodes in the network. The nodes in my dynamic network will create these edges when the load is distributed using the random sampling protocol. Thus, the network is a truly self-organised dynamic system. To effectively distribute the load, it is required to keep the network in a correct state so that the BRS load-balancing scheme would work properly. Yet, it is easy to keep the network in a correct state by using local edge connecting procedures. Therefore, if a link is failed, it can easily be fixed without affecting the system performance.

The network state maintenance and workload allocation in the BRS scheme are based only on the local information and processes. That is, each node decides the amount of resource or computing power it needs to share, and it integrates this information into the network structure through the random sampling protocol. In the same way, new load is distributed based on only the information available via local investigation of the network.

The BRS scheme produces dynamic random graph topologies, and the resulting network will be resilient to random faults and does not have a central point of failure. Moreover, the BRS load-balancing algorithm does not require the nodes to be aware of each other which allow the BRS scheme to scale well to very large network sizes. Accordingly, the generated network system provides an effective, reliable, and scalable load-balancing paradigm for the distributed resources accessible on large-scale networks.

A statistical mechanical model for load-balancing paradigm based on Complex networks theory is developed. An analytical solution for the stationary distribution of node's degree in the network is derived. The steady state analysis of node's in-degree distribution confirms that the load distribution of the generated network system is compatible with ER random networks. Yet, since every node in the network should have the same in-degree, the optimal degree distribution would be a regular graph as the nodes would have the same load. Therefore, the performance of the proposed load-balancing technique is enhanced to generate more regular graphs.

Extensive simulations results have been used to evaluate the performance and to measure the efficiency of the proposed load-balancing scheme. The performance of the proposed load-balancing technique is examined for both homogeneous and heterogeneous configurations. In heterogeneous configuration, the nodes have different capabilities and resources with Power-law resource (in-degree) distributions. Moreover, I took into account the effect of communication delay on the performance of the proposed load-balancing technique. Simulation results agreed with the analytical predictions for the stationary in-degree distributions and confirm that the resulted network system can provide an effective load-balancing paradigm for the distributed resources accessible on large-scale networks.

One of the strong points of the proposed algorithm is that a random sampling of $\log(N)$ steps is sufficient to reach an in-degree variance very close to the optimum variance. Furthermore, the time required to efficiently balance the load distribution in the network has a logarithmic relation with the size of the network. In view of that, the proposed BRS technique is a very efficient load balancing algorithm and suitable for growing networks.

The throughput and bandwidth capacity performance of the BRS scheme are investigated and compared it with the centralised scheme. I observed that the BRS load-balancing scheme reduced the bandwidth capacity consumed by an individual node in the network compared to that within the centralised scheme. Besides, I observed the total throughput for BRS algorithm is close to the total throughput of the centralised algorithm, while my scheme has the advantage of being a distributed scheme. Then, I showed that introducing geographic awareness factor in the random

sampling has noticeably reduced the effects of communication latency in Grid Network environments.

The main achievements of the proposed load distribution algorithm can be summarised in the following points:

- BRS scheme is a straightforward and can easily be implemented scheme using standard networking protocols. The decentralised feature of this scheme makes it suitable for many applications involving large number of nodes such as Grid Networks.
- The BRS scheme provides a decentralised and scalable load balancing performance. I have proved that the network structure converges to a regular graph under ideal conditions, which gives us load balancing.
- The BRS scheme does not require the nodes to be aware of one another which allow the BRS algorithm to scale well to very large system sizes.
- The network maintenance and job allocation protocols are based only on local information and actions. Thus, the BRS scheme is a truly self-organized dynamic system.
- Only $\log(N)$ random sampling steps are sufficient for BRS algorithm to efficiently distribute and balance the workload. Thus, the overhead incurred from the proposed scheme is minimal.
- The BRS scheme produces network system that is resilient to random faults since it does not have a central point of failure and it can dynamically heal itself against node failure or attack.
- The time required for BRS algorithm to balance the load distribution has a logarithmic relation with the size of the network.

- Analytical and simulation results confirm that the BRS protocol can effectively distribute the load in both homogeneous and heterogeneous large-scale networks.
- The bandwidth capacity and throughput performance achieved by the BRS scheme is close to the performance of the ideal centralised scheme.
- Finally, introducing latency optimising factor in the random sampling has noticeably reduced the effects of communication latency in Grid Network environments.

To conclude, the proposed biased random sampling (BRS) load-balancing paradigm is a novel technique for any type of resource sharing and it is not limited to the distributed computing environment. The in-degree of a node can be made to correspond to any type of shareable resources. And since the generated networks have low diameters, this makes them easy to sample using random sampling. Analytical and simulation results confirm the efficiency of this approach in networks with a large range of resource and load distributions. Thus, the proposed load-balancing algorithm generates network systems which are scalable, self-organised, robust, and depend only on local information for load distribution and resource discovery.

The research work described in this thesis has resulted in Journals and international conferences publications, which are included in the list of references (Rahmeh, et al., 2008; Rahmeh, et al., 2007; Rahmeh and Johnson, 2008; Rahmeh, Johnson, and Lehmann, 2007; Rahmeh and Johnson, 2007; Rahmeh, et al., 2006)."

7.2 Further Work

A number of potential improvements to my load distribution technique and generalisations of my model deserve further study. This work can be extended to include cases where jobs may require certain quality of services (QoS); such as communications bounded, distance sensitive, and time bounded services. In addition, extra work is required to study the case where jobs depend on the output of other jobs. This will help us in understanding how these situations will affect the nodes' in-degree and load distribution in the network. Examining how these considerations will affect the efficiency of the load balancing could be a topic for future work.

In this thesis, an analytical expression for the steady-state in-degree distribution using random sampling that selects the last node in sampling to give it the new load is derived, and it is shown that it produces ER random graphs. I further improved the algorithm to use a biased random sampling which selects the highest in-degree node rather than the last node. I have shown by using extensive simulations that the algorithm generates nearly regular random graphs. Theoretical analysis of the biased random sampling that selects the highest in-degree will be a useful extension for future work.

The biased random sampling is proved to be an efficient technique to distribute the load. However, this efficiency comes at an extra overhead which may increase the communication delay overhead. To decrease the communication delay overhead, the number of random samplings initiated by each node could be increased. That is, instead of just sending out one random sampling, a requesting node sends j random samplings, and each one takes its own random walk. The expectation is that j

random samplings after ℓ_{RS} steps should reach approximately the same number of nodes as one random sampling after $j\ell_{RS}$ steps. Therefore, by using j random samplings, it is expected to cut the delay down by a factor of j. Examining how this technique will affect the load-balancing efficiency can be considered for future work.

The formal structure of Grid Networks is expected to represent patterns of communication and organization, and to influence the nature of communication in these networks. Studying and analyzing the topology of the Grid is extremely important for deriving the mathematical model that describes the dynamics on Grid Networks. Dynamics on networks refer to dynamic processes that take place in a network topology. Therefore, there is a need to define the degree distribution that describes the *growing* behaviour of Grid Networks, average and shortest path length between two nodes, clustering coefficient, node types (e.g., single or bipartite (Newman et al., 2001), and other networks characteristics. It is expected that they will display characteristics that are different from small-world or scale-free networks.

Despite the fact that several types of complex networks which are of high interest for the scientific community, such as the Web, cellular networks, Internet, and some social networks follow power-law distribution, Grid Networks are expected to have an exponential distribution. Furthermore, due to its dynamic and adaptive behaviour, Grid Networks can develop both power-law and exponential degree distributions. Thus, if all processes shaping the topology of the network are properly incorporated, the resulting distribution could have a complex form, described by a combination of power-laws and exponentials distribution.

References

Abramowitz, M. & Stegun, I. A. (1972) *Handbook of Mathematical Functions with formulas, Graphs, and Mathematical Tables*. Dover Publications, 9th Edition, New York.

Adabala, S., Chadha, V., Chawla, P., Figueiredo, R., fortes, J., et al. (2005) From Virtualized Resources to Virtual Computing Grids: the In-Vigo System. *Future Generation Computer Systems*, 21(6), 896-909.

Adamic, L.A. (1999) The Small World Web. *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*, ECDL, 1696, 443-452.

Albert R., Jeong H. & Barabási A. L. (1999) Diameter of the World Wide Web. *Nature*, 401(6749), 130-131.

Albert, R. & Barabási, A. L. (2002) Statistical Mechanics of Complex Networks. *Reviews of Modern Physics*, 74(1), 47-97.

Albert, R., Jeong, H., & Barabasi, A. L. (2000) Attack and Error Tolerance Of Complex Networks, *Nature*, Vol. 406, pp. 378-382

Amaral, L. A. N., Scala, A., Barthelemy, M. & Stanley, H. E. (2000) Classes of Behavior of Small-World Networks. *Proceedings of the National Academy of Sciences*, 97(21), 11149-11152. Arxiv: Cond-Mat/0001458.

Andreolini, M.Colajanni, M., & Morselli, R. (2002) Performance Study of Dispatching Algorithms in Multi-Tier Web Architectures, *ACM SIGMETRICS Performance Evaluation Rev.*, vol. 30, no. 2, pp.10-20, ISSN 0163-5999.

Avin, C. & Brito, C. (2004) Efficient and Robust Query Processing in Dynamic Environments Using Random Walk Techniques, *Proceedings of the Third international Symposium on information Processing in Sensor Networks*. ACM Press, 277-286.

Barabási, A. L. & Albert, R. (1999) Emergence of Scaling in Random Networks. *Science*, 286, 509-512.

Barabási, A. L. (2001) The Physics of the Web. *Physics World*. 14(7), 33-38.

Barabási, A. L. & Frangos, J. (2002) *Linked: The New Science of Networks*, Perseus Books, Cambridge, MA, USA, ISBN: 978-0738206677.

Barabási, A. L., Dezso, Z., Ravasz, E., Yook, S., and Oltvai, Z. (2003) Scale-Free and Hierarchical Structures in Complex Networks, AIP Conference Proceedings, Volume 661, Issue 1, pp. 1-16.

Barabási, A. L., & Oltvai, Z. N. (2004) Network Biology: Understanding the Cell's Functional Organization, *Nature Reviews Genetics 5*, pp. 101-113.

Barrat, A. & Weigt, M. (2000) On the Properties of Small-World Network Models. *The European Physical Journal*, 13, 547-560.

Becker, W. (1995) Dynamic Balancing Complex Workload in Workstation Networks - Challenge, Concepts and Experience. *High Performance Computing and Networking (HPCN95)*, Milan, Italy, Springer, pp. 407-412.

Becker, W. & Waldmann, G. (1994) Exploiting Inter Task Dependencies for Dynamic Load Balancing. *IEEE 3rd Int. Symposium on High Performance Distributed Computing*. San Francisco, California.

Becker, W. & Waldmann, G. (1995) Adaptation in Dynamic Load Balancing: Potential and Techniques. *Fachtagung Arbeitsplatz-Rechensysteme*, Hanover, Germany.

Becker, W. & Zedelmayr, J. (1994) Scalability and Potential for Optimization in Dynamic Load Balancing: Centralized And Distributed Structures. *Parallel Algorithmen und Rechnerstrukturen*, Potsdam.

Blair, G.S., F. Costa, G. Coulson, H. Duran, et al. (1999) The Design of a Resource-Aware Reflective Middleware Architecture, *Proceedings of the 2nd international Conference on Meta-Level Architectures and Reflection*, St. Malo, France.

Bollabás, B. (2001) *Random Graphs*. Second Edition, Cambridge, Cambridge University Press.

Bollobás, B. (1985) *Random Graphs*. Academic Press, London, England.

Bornholdt, S. & Schuster, H. G. (2003) *Handbook of Graphs and Networks: From the Genome to the Internet*. Weinheim, Wiley-VCH.

Braun, T. D., et al. (2001) A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. *Journal of Parallel and Distributing Computing*, 61(6):810–837.

Buchanan, M. (2002) *Nexus: Small Worlds and the Groundbreaking Science of Networks*. New York, London, W.W. Norton & Company.

Bustos, J., & Caromel, D. (2006) Load Balancing: Toward the Infinite Network. *12th Workshop on Job Scheduling Strategies for Parallel Processing*, France.

Callaway, D. S., Newman, M. E. J., Strogatz, S. H. & Watts, D. J. (2000) Robustness and Fragility: Percolation on Random Graphs. *Physical Review Letter*, 85, 5468-5471.

Cardellini, V.Casalicchio, E.Colajanni, M., & Yu, P. S. (2002) The State Of The Art In Locally Distributed Web-Server Systems', *ACM Computing Surveys*, Vol. 34, No. 2, pp. 263-311, ISSN 0360-0300.

Casavant, T. & Kuhl, J. (1988) A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems. *IEEE Transactions on Software Engineering*, 14(2), 141-154.

Casavant, T., & Kuhl, J. (1988) Effects of response and stability on scheduling in distributed computing systems. *IEEE Transactions on Software Engineering*, pp. 1578-1588.

Castillo, C. (2004) Effective Web Crawling, *PhD Thesis*, University of Chile.

Coulouris, G., Dollimore, J. & Kindberg, T. (2001) *Distributed Systems: Concepts and Design*. Addison-Wesley, Pearson Education.

Cvetković, D. M., Doob, M., and Sachs, H. (1998) *Spectra of Graphs: Theory and Applications*, 3rd edition, New York, Wiley, ISBN: 978-3527296859.

Czajkowski, K., Fitzgerald, S., Foster, I., and Kesselman, C. (2001) Grid Information Services for Distributed Resource Sharing. Proc. 10th IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press.

Dorogovtsev, S. N. & Mendes, J. F. F. (2002) Evolution of Networks. *Advances in Physics*, 51(4), 1079-1187.

Dorogovtsev, S. N. & Mendes, J. F. F. (2003). *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, Oxford.

Dorogovtsev, S. N., Mendes, J. F. F. & Samukhin, A. N. (2000). WWW and Internet Models From 1955 Till Our Days and the "Popularity Is Attractive" Principle. *Physics E-Print Archive*: Arxiv: Cond-Mat/0009090.

Drougas, Y., Repantis, T., & Kalogeraki, V. (2006) Load Balancing Techniques for Distributed Stream. *Processing Applications in Overlay Environments*. ISORC'06, USA.

Eager, D. L., Lazowska, E. D., & Zahorjan, J. (1985) A Comparison Of Receiver-Initiated and Sender-Initiated Adaptive Load Sharing. *Performance Evaluation*, 6(1):53–68.

Eager, D. Lazowska, E., & Zahorjan, J. (1986) Adaptive Load Sharing in Homogeneous Distributed Systems. *IEEE Transactions on Software Engineering*, pp. 662-675.

Els, R. & Monien, B. (2003) Load Balancing of Unit Size tokens and Expansion Properties of Graphs. *Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures*. ACM Press, 266–273.

Erdös, P. & Rényi, A. (1959) *On Random Graphs*. Publicationes Mathematicae, 6, 290-297.

Erdös, P. & Rényi, A. (1960) On the Evolution of Random Graphs. *Publications of the Mathematical institute of the Hungarian Academy of Sciences*, 5, 17-61.

Erdös, P. & Rényi, A. (1961) On the Evolution of Random Graphs. *Bull. Inst. Internat. Statistics*, 38, Tokyo, 343-347.

Faloutsos, M., Faloutsos, P. & Faloutsos, C. (1999) On Power-Law Relationships of the Internet Topology. *Proceedings of ACM/SIGCOMM, Computer Communication Review,* Cambridge, MA, *29*(4), 251-262.

Favarim, F., Siqueira, F., & Fraga, J. (2003) Fault-Tolerant CORBA Components. The 2nd Workshop on Reflective and Adaptive Middleware. *Proceedings of the Middleware*, Rio De Janeiro, Brazil.

Feng, Y.Li, D., Wu, H., & Zhang, Y. (2000) A dynamic load balancing algorithm based on distributed database system. *Proc. 4th Intl. Conf. on High Performance Computing in the Asia-Pacific Region*, pages 949–952, Beijing, China.

Ferrari, D., & Zhou, S. (1987) An Empirical Investigation Of Load Indices For Load Balancing Applications. *Technical report*, University of California, Berkeley.

Foster, I. & Kesselman, K. (1999) *The Grid: Blueprint for A Future Computing Infrastructure*. Morgan Kaufmann.

Foster, I., Kesselman, C., Nick, J. & Tuecke, S. (2002) The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. *Argonne National Laboratory*, Technical Report, Chicago, USA.

Freeman, L.C. (1979) Centrality in Networks: Conceptual Clarification. *Social Networks,* Vol. 1, 215–239.

Frey, J., Tannenbaum, T., Foster, I., Livny, M., Tuecke, S. (2001) Condor-G: A Computation Management Agent for Multi-Institutional Grids, In Proceedings of 10th International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press.

Ganek, A. & Corbi, T. A. (2003) The Dawning of the Autonomic Computing Era. *IBM Systems Journal*, 42(1).

Globus (2003) MDS 2.2 User's Guide. *The Globus Project*, [Online] www.Globus.Org/toolkit/Docs/2.4/Mds/Mdsusersguide.Pdf.

Goh, K. I., Oh, E., Jeong, H., Kahng, B. & Kim, D. (2002) Classification of Scale-Free Networks. *Proceedings of the National Academy of Sciences*, 99(20), 12583-12588.

Gustafson, J. (1988) Reevaluating Amdahl's Law. *Communications of the ACM*, pp. 532-533.

Handcock, M. S., & Jones, J. H. (2003) An Assessment Of Preferential Attachment As A Mechanism For The Growth Of Human Sexual Networks, *Proceedings of the Royal Society*, B 270, 1123 -1128.

Hayes, B. (2000) Graph Theory in Practice: Part II. *American Scientist*, 88(2), 104-109.

Holme, P., Kim, B. J., Yoon, C. N. & Han, S. K. (2002) Attack Vulnerability of Complex Networks. *Physical Review E*, 65(5), Art-No.056109.

Horman, S. (2001) Super Sparrow: Global Load Balancing Solution for Linux. *URL http://www.supersparrow.org/.*

Horman, S. (2005) Ultra Monkey 3: High Availability and Load Balancing Solution for Linux. *URL http://www.ultramonkey.org/.*

Huberman, B. A. (2001) *The Laws of the Web: Patterns in the Ecology of information.* The MIT Press, Cambridge, MA.

Huberman, B. A., & Adamic, L. A. (1999) Growth Dynamics of the World Wide Web. *Nature, 401*(6749), 131.

Jennings, N.R. (2000) On Agent-Based Software Engineering. *Artificial Intelligence*, 177(2), 277-296.

Jeong, H., Tombor, B., Albert, R., Oltvai, ZN. & Barabsi, A. L. (2000) The Large-Scale Organization of Metabolic Networks. *Nature*, 407, 651-654.

Jin, E. M.Girvan, M., & Newman, M. E. J. (2001) The Structure Of Growing Social Networks, *Phys. Rev.*, Ed. 64, pp. 381-399.

Kalé, L. V. (1988) Comparing the performance of two dynamic load distribution methods. *Proc. Intl. Conf. on Parallel Processing*, pages 77–80.

Kephart, J. O. & Chess, D. M. (2003) The Vision of Autonomic Computing. *IEEE Computer*, 36(1), 41-50.

Kleinrock, L. (1975) *Queueing Systems. Volume I: Theory.* John Wiley & Sons, New York.

Krapivsky, P. L. & Redner, S. (2002) A Statistical Physics Perspective on Web Growth. *Computer Networks*, 39(3), 261-276.

Kremien, O. & Kramer, J. (1992) Methodical Analysis of Adaptive Load Sharing Algorithms. *IEEE Transactions on Parallel Distribution Systems*, 3(6), 747–760.

Kropf, P. (1996) Load Balancing: Short Course On Advanced Parallel Computation. *COSMASE-EPFL-Lausanne*, Suisse.

Kumar, V., Grama, A., & Rao, V. (1994) Scalable load balancing techniques for parallel computers. *Journal of Parallel and Distributed Computing*, 22(1):60–79.

Kunz, T. (1991) The influence of different workload descriptions on a heuristic load balancing scheme. *IEEE Transactions on Software Engineering*, pp. 1327-1341.

Laue, R. (1970) *Elements of Graph Theory and its Applications in Biological Sciences*. Academic Publishing House, Geest and Portig, K.G., Leipzig, Germany.

Lehmann, S., Jackson, A. D., & Lautrup, B. (2005) Life, death and preferential attachment. *Europhysics Letters*.

Lin, H., & Raghavendra, C. (1992) A dynamic load-balancing policy with a central job dispatcher (LBC). *IEEE Trans. Software Engineering*, 18(2):148–158.

Litzkow, M., Livny, M., & Mutka, M. (1988) Condor: A Hunter of Idle Workstations. *Proceedings of the Eighth International Conference of Distributed Computing Systems*.

Loeve, M. (1977) Probability Theory, *Graduate Texts in Mathematics*, Volume 45, 4th edition, Springer-Verlag.

Lov'asz, L. & Winkler, P. (1995) Mixing of Random Walks and Other Diffusions on a Graph. *Surveys in Combinatorics*, London Mathematical Society Lecture Note Series, Cambridge University Press, 218, 119-154.

Lüling, R. & Monien, B. (1993) A Dynamic Distributed Load Balancing Algorithm with Provable Good Performance. *SPAA '93: Proceedings of the Fifth Annual ACM Symposium on Parallel Algorithms and Architectures*, ACM Press, New York, USA, 164–172.

Lüling, R.Monien, B., & Ramme, F. (1991) A Study of Dynamic Load Balancing Algorithms. *Proceedings of the 3rd IEEE SPDP*, 686-689.

Maxemchuk, N. & El-Zarki, M. (1990) Routing and Flow Control in High-Speed Networks. *Proceedings of IEEE*, 78(1), 204-221.

Mengotti, T. (2004) GPU, a Framework for Distributed Computing over Gnutella. ETH Zurich, Master Thesis, Switzerland.

Mengotti, T., Petersen, W. P. and Arbenz, P. (2002) Distributed Computing Over Internet Using A Peer To Peer Network, Zurich, Switzerland.

Milgram, S. (1967) The Small World Problem. *Psychology Today*, 2, 60-67.

Mitzenmacher, M. (2001) The Power of Two Choices in Randomized Load Balancing. *IEEE Transactions on Parallel Distribution Systems*, 12(10), 1094–1104.

Mitzenmacher, M. (2003) A Brief History of Generative Models for Power Law and Lognormal Distributions. *Internet Math.*, Vol. 1, No. 2, pp. 226-251.

Monien, B. (1996) Load Balancing Driven Process Migration. *EURO-PAR'96*, Lyon, France.

Montresor, A., Meling, H. & Babaoglu, O. (2002) Messor: Load-Balancing Through a Swarm of Autonomous Agents. *Proceedings 1st International Workshop on Agents and P2P Computing*, Bologna, Italy.

Motwani, R. & Raghavan, P. (1995) *Randomized Algorithms*. Cambridge University Press.

Murata, Y., et al. (2006) A Distributed And Cooperative Load Balancing Mechanism For Large-Scale P2P Systems. *SAINT-W*, USA.

Newman, M. E. J. (2000) Models of the Small World: A Review. *Journal of Statistical Physics*, 101(3-4), 819-841.

Newman, M. E. J. (2001) Scientific Collaboration Networks: I. Network Construction and Fundamental Results. *Physical Review*, 64(1), Art-No. 016131.

Newman, M. E. J., & J. Park, J. (2003) Why Social Networks are Different from Other Types of Networks, *Phys. Rev.*, Ed. 68(3).

Newman, M. E. J., Strogatz, S. H. & Watts, D. J. (2001) Random Graphs With Arbitrary Degree Distributions and their Applications. *Physical Review*, 64 (026118).

Ngg2 Group (2004) Next Generation Grids 2: Requirements and Options for European Grids Research 2005-2010 and Beyond. *Expert Group Report.*

Oppenheimer, D., Albrecht, J., Patterson, D. & Vahdat, A. (2004) Scalable Wide-Area Resource Discovery. *Technical Report CSD04-1334*, University of California Berkeley, Berkeley, CA, USA.

Ozden, B. Goldberg, A., & Silberschatz, A. (1993) Scalable and non-intrusive load-sharing in owner-based distributed systems. *The 5th IEEE Symposium on Parallel and Distributed Processing*, Dallas, pp. 690-699.

Peixoto, L. P. (1996) Load Distribution: A Survey. *Technical Report.* Departamento De inform'atica, Escola De Engenharia, Universidade Do Minho.

Peterson, L., Anderson, T., Culler, D., & Roscoe, T. (2002) A Blueprint for Introducing Disruptive Technology into the Internet. *Proceedings of the First ACM Workshop on Hot Topics in Networking (HotNets)*.

PingER (2005) PingER: The Internet End-to-end Performance Measurement (IEPM) project. URL: http://www-iepm.slac.stanford.edu/pinger/.

Pennock, D., Flake, G. W., Lawrence, S., Glover, E., & Giles, C. L. (2002) Winners don't take all: Characterizing the competition for links on the Web, *Proceedings of the National Academy of Sciences*, 99 (8): 5207-5211.

Rahmeh, O. A., Johnson, P., and Taleb-Bendiab, A. (2008) A Dynamic Biased Random Sampling Scheme for Scalable and Reliable Grid Networks. *The INFOCOMP Journal of Computer Science*, Vol. 7, No. 4, ISSN: 1807-4545, Page(s): 1-10.

Rahmeh, O. A., Johnson, P., and Lehmann, S. (2007) A Fitted Random Sampling Scheme for Load Distribution in Grid Networks. *The International Journal of Information Technology*, V4, No. 2, ISSN: 2070-3724, Page(s): 153-158.

Rahmeh, O. A., Johnson, P. (2008) Towards Scalable and Reliable Grid Networks. *The IEEE/ACS International Conference on Computer Systems and Applications*, ISBN: 978-1-4244-1967-8, Page(s): 253–259, Doha, Qatar.

Rahmeh, O. A., Johnson, P., and Lehmann, S. (2007) A Fitted Random Sampling Scheme for Load Distribution in Grid Networks. *Proceedings of World Academy of Science, Engineering and Technology*, ISSN: 1307-6884, ISSN: 2070-3724, Page(s): 204-209, Nice, France.

Rahmeh, O. A., Johnson, P. (2007) A Distributed Load Balancing Mechanism for Large-Scale Networks. *The International Conference on Communications and Computer Applications and Technologies (CCAT 2007)*, Amman, Jordan.

Rahmeh, O. A., Johnson, P., and Taleb-Bendiab, A. (2006) A Decentralized Load Balancer for Grid Networks. *The SPIE's Optics East 2006*, Boston, Massachusetts, USA.

Rapoport, A. (1957) Contribution to the Theory of Random and Biased Nets. *Bulletin of Mathematical Biophysics*, 19, 257-277.

Riedl, R., & Richter, L. (1996) Classification of load distribution algorithms, *The 4th Euro-Micro Workshop on Parallel and Distributed Processing*, Braga, Portugal, IEEE Computer Society Press, pp 404-413.

Sarshar, N., Boykin, P. O., & Roychowdhury, V. (2004) Percolation Search in Power Law Networks: Making Unstructured Peer-to-Peer Networks Scalable, *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, pp. 2–9.

Sarshar, N., & Roychowdhury, V. (2004) Scale-Free and Stable Structures in Complex Ad-Hoc Networks, *Physical Review E.*, v69, 026101.

Schantz, R. E. & Schmidt, D. C. (2001) Middleware for Distributed Systems: Evolving the Common Structure for Network-Centric Applications, *Encyclopaedia of Software Engineering*, Wiley & Sons, New York.

Scharnhorst, A. (2003) Complex Networks and the Web: Insights from Nonlinear Physics, *Journal of Computer-Mediated Communication*, Vol. 8, No. 4.

Scheurer, C., Scheurer, H., & Kropf, P. (1995) Load balancing driven process migration. *Technical report*, University of Berne.

Scott, J. (2001) *Social Network Analysis: A Handbook.* Sage Publications, London.

Servetto, S. D. & Barrenechea, G. (2002) Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks. *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, 12-21.

Shin, K. G., & Chang, Y. C. (1995) A coordinated location policy for load sharing in hypercube connected multicomputers. *IEEE Transactions on Computers*, pp. 669-682.

Shivaratri, N.Krueger, P., & Singhal, M. (1992) Load distributing for locally distributed systems. *IEEE Computer*, 25(12):33–44.

Slanina, F. & Kotrla, M. (2000) Random Networks Created By Biological Evolution. *Physical Review*, E62, pp. 6170-7177.

Slanina, F. & Kotrla, M. (1999) Extremal Dynamics Model on Evolving Networks. *Physical Review Letter,* E83, pp. 5587-5590.

Solomonoff, R. & Rapoport, A. (1951) Connectivity of Random Nets. *Bulletin of Mathematical Biophysics*, 13, 107-117.

Stankovic, J. (1985) Stability and distributed scheduling algorithms. *IEEE Transactions on Software Engineering*, pp. 1141-1152.

Strogatz, S. H. (2001) Exploring Complex Networks. *Nature*, 410(6825), 268-276.

Subramanian, R. & Scherson, I. D. (1994) An Analysis of Diffusive Load Balancing. *Proceedings of the Sixth Annual ACM Symposium on Parallel Algorithms and Architectures*, ACM Press, 220–225.

Tadic, B. (2001) Dynamics of Directed Graphs: the World Wide Web. *Physica A, 293*(1-2), 273-284.

Taylor, I. J. & Andrew Harrison, A. (2005) From *P2P to Web Services and Grids: Peers in a Client/Server World.* Springer, ISBN: 978-1852338695.

Theimer, M., & Lantz, K. (1989) Finding idle machines in a workstation based distributed system. *IEEE Transactions on Software Engineering*, 15(11), pp. 1444-1458.

Tian, Hi, Shen, H., & Matsuzawa, T. (2005) Random walk routing for wireless sensor networks. *Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, USA.

Torrellas, J., Tucker, A., & Gupta, A. (1995) Evaluating the performance of cache-affinity scheduling in shared-memory multiprocessors. *Journal of Parallel and Distributed Computing*, 24(2):139–151.

Tzen, T., & Ni, L. (1993) Trapezoid self-scheduling: A practical scheduling scheme for parallel computers. *IEEE Trans. Parallel and Distributed Systems*, 4(1):87–98.

Vázquez, A., Pastor-Satorras, R. & Vespignani, A. (2002) Internet Topology at the Router and Autonomous System Level. *Physics E-Print Archive*, Arxiv: Cond-Mat/0206084.

Wang, F. X. (2002) Complex Networks: Topology, Dynamics and Synchronization. *International Journal of Bifurcation and Chaos*, 12(5), 885-916.

Watts, D. J. & Strogatz, S. H. (1998) Collective Dynamics of 'Small World' Networks. *Nature*, 393(6684), 440-442.

Watts, D. J. (1999) *Small Worlds: The Dynamics of Networks between Order and Randomness*, Princeton, New Jersey, Princeton University Press.

Weiss, G. (1999) *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press.

Willebeek-LeMair, M., & Reeves, A. (1993) Strategies For Dynamic Load Balancing On Highly Parallel Computers, *IEEE Trans. Parallel and Distributed Systems*, 9(4):979–993.

Wolf, J. L., & Yu, P. S. (2001) On Balancing The Load In A Clustered Web Farm, *ACM Transactions on Internet Technology*, pp. 231-261, ISSN 1533-5399.

Xu, C. Lüling, R.Monien, B., & Lau, F. C. M. (1995) An Analytical Comparison Of Nearest Neighbours Algorithms For Load Balancing In Parallel Computers, *The 9th International Parallel Processing Symposium*, Paderborn, Germany.

Yagoubi, B., & Slimani, Y. (2007) Task Load Balancing Strategy for Grid Computing. *Journal of Computer Science*, 3 (3) 186-194.

Zhang, Wensong, & Zhang, Wenzhuo (2003) Linux Virtual Server Clusters: Build highly-scalable and highly-available network services at low cost. *Linux Magazine*.

Zhou, S. (1988) A trace-driven simulation study of dynamic load balancing. *IEEE Trans. Software Engineering*, 14(9):1327–1341.

Appendix

List of Authors' Publications

Journals Publications

Rahmeh, O. A., Johnson, P., and Taleb-Bendiab, A. (2008) A Dynamic Biased Random Sampling Scheme for Scalable and Reliable Grid Networks. *The INFOCOMP Journal of Computer Science*, Vol. 7, No. 4, ISSN: 1807-4545, Page(s): 1-10.

Rahmeh, O. A., Johnson, P., and Lehmann, S. (2007) A Fitted Random Sampling Scheme for Load Distribution in Grid Networks. *The International Journal of Information Technology*, V4, No. 2, ISSN: 2070-3724, Page(s): 153-158.

International Conferences Publications

Rahmeh, O. A., Johnson, P. (2008) Towards scalable and reliable Grid Networks. *The IEEE/ACS International Conference on Computer Systems and Applications*, ISBN: 978-1-4244-1967-8, Page(s): 253–259, Doha, Qatar.

Rahmeh, O. A., Johnson, P., and Lehmann, S. (2007) A Fitted Random Sampling Scheme for Load Distribution in Grid Networks. *Proceedings of World Academy of Science, Engineering and Technology*, ISSN: 1307-6884, ISSN: 2070-3724, Page(s): 204-209, Nice, France.

Rahmeh, O. A., Johnson, P. (2007) A Distributed Load Balancing Mechanism for Large-Scale Networks. *The International Conference on Communications and Computer Applications and Technologies (CCAT 2007)*, Amman, Jordan.

Rahmeh, O. A., Johnson, P., and Taleb-Bendiab, A. (2006) A Decentralized Load Balancer for Grid Networks. *The SPIE's Optics East 2006*, Boston, Massachusetts, USA.