Inverse shortest path algorithm for weighted graphs

Flow-based resource management for path networks

ET4300: Master Thesis

Sai Poojitha Darsi (5024749)

Delft University of Technology

Inverse shortest path algorithm for weighted graphs

Flow-based resource management for path networks

by

Sai Poojitha Darsi (5024749)

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday July 22, 2021 at 11:00 AM.

Responsible Full Professor:	Prof.dr.ir P.F.A. Van Mieghem
Daily Supervisor:	Ir. R.A.C.J. Noldus
Institution:	Delft University of Technology
Place:	Faculty of Electrical Engineering, Mathematics and Computer Science, Delft
Project Duration:	1 December, 2020 - 22 July, 2021

Cover Image: www.vocowallpaper.com

TUDelft

Preface

This thesis marks the completion of my Masters study in TU Delft at Network Architecture and Services (NAS) group of Electrical Engineering, Mathematics and Computer Science (EEMCS) faculty. The two year study in Delft has been a great learning experience. I would like to thank Prof. Piet Van Mieghem and Ir. Rogier Noldus for their constant guidance and support throughout the thesis. I feel fortunate to have worked under their supervision and to witness their ways of working in research. My sincere gratitude to Prof. Piet Van Mieghem for giving me the opportunity to work on an exciting and challenging project and his contribution in showing the path by posing the right questions at right time. I am thankful to my supervisor Rogier Noldus, for the guidance and brainstorming sessions in weekly progress meetings and his feedback for the report and presentations has helped me to improve my work a lot. I would like to thank NAS group especially Bastian Prasse, Gabriel Budel and Peng Sun for all the timely help with setting up QCE cluster account to run the simulations and codes. I would like to thank Karel Devriendt for the clarifications and support provided over emails. Additionally, I would like to thank Dr. Sebastian Feld for being part of the thesis committee.

I am thankful to my parents for believing in me and supporting me even during the challenging times. I thank Saran for being there at all times and for the motivation and support. I would like to thank Sylvia for making me realize life is beautiful and simple. I am thankful to God for having taught me ”Whatever is seeking you is all helping you”. One of many thoughts that kept coming to me during this period of study - ”As a rock loses its roughness and sharpness as it flows through the river to become a pebble as it is its destiny, enjoy this process of transformation to realize *you* and your destiny.”

Sai Poojitha Darsi (5024749)
Delft, July 2021

Abstract

With the proliferating networks, resource allocation based on Quality of Service (QoS) constraints mapping has been one of the difficulties faced by Internet Service Providers (ISP). The advent of new technologies such as virtualization and cloud computing have enabled the users to access content from anywhere around the world. This results in a need for an efficient and fast resource allocation method based on demand acting on the network. Many researchers have developed various algorithms to address this issue. However, they have considered only flow-based network properties, while the contemporary networks communicate mostly through path based communication using the shortest paths. Inverse shortest path algorithm (ISPA) is a graph-theory based heuristic developed to solve this network resource allocation problem which considers both flow-based communication properties through Effective resistance matrix and path based communication properties through Shortest path matrix. Nevertheless, ISPA is so far implemented only for unweighted graphs. This study aims at assessing the feasibility of ISPA for weighted graphs by understanding the nature of Inverse shortest path problem (ISPP) bounds in weighted random graphs and implementing ISPA for weighted graphs. ISPP bounds behaviour is studied for weighted graphs by analyses of Q-norm distributions, probability of failure distributions and hopcount distributions. A feasibility condition verifying ISPP bounds is derived in relation with the input parameters of the weighted random graph. The solutions obtained through hop count distribution analysis are observed to be Poissonian in nature. Finally, ISPA is implemented for weighted graphs such that demand matrix can be resolved into a simple distance matrix to obtain solution which is a non-negative weighted Adjacency matrix and feasibility conditions are verified for the solutions obtained through ISPA.

Keywords: Resource allocation, inverse shortest path problem, shortest path computations.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
QoS	Quality of Service
ISP	Internet Service Provider
IP	Internet Protocol
TE	Traffic Engineering
ISPP	Inverse shortest path problem
ISPA	Inverse shortest path algorithm
ER	Erdös Rényi

Symbols

Symbol	Definition
G	Graph
N	Number of nodes
p	Link density
n	Set of nodes
L	Number of links
l	Set of links
r	Average link resistance
r_0	Critical threshold
c	Slope factor
D	Demand matrix
L_{max}	Maximum number of links
$G_p(N)$	Binomial model representation of ER graph
$E[L]$	Average number of links
R^n	Euclidean space
A	Adjacency matrix
a_{ij}	Link existence between node i and j
$\tilde{A}$	Weighted Adjacency matrix
d_i	Degree of node i
d_{avg}	Average degree of the graph
u	All one vector
d	Degree vector
S	Shortest path matrix
P_{ij}	Path from node i to j
w_l	Weight of the link l
$P^*_{i\to j}$	Shortest path from node i to j
H_{ij}	Hopcount between nodepair (i, j)
Ω	Effective resistance matrix
Q	Laplacian matrix
μ_k	k_{th} Eigen value of Laplacian
z_k	Eigen vector corresponding to μ_k
$Q^\dagger$	Pseudo Inverse of Q
ζ	Diagonal vector of $Q^\dagger$
R_G	Effective graph resistance
w_{ab}	Weight of the link between nodepair (a, b)
$\tilde{\Omega}$	Weighted effective resistance matrix
$\tilde{R_G}$	Weighted effective graph resistance
Δ	Degree matrix

List of Figures

List of Tables

1

Introduction

Every network with its many links, has a certain demand acting on each node pair which can be quantified through bandwidth requirements and QoS requirements such as end-to-end delay, packet-loss, and jitter etc. Bandwidth and buffer capacity is generally allocated based on the demand acting on the network rather than the service provisioning in the conventional packet networks by Internet Service Providers (ISPs). With an increase in high-speed communications, to meet the network demands, over-provisioning of fast speed routing devices and high-speed network links was performed, which resulted in cheaper and faster internet-based connectivity [1].

With this proliferation of IP networks, the number of services and data on the networks has been increasing considerably which has laid the path for Traffic Engineering (TE), a set of network management tools and guidelines for performance optimization and evaluation in the operational IP networks [2]. But, resource allocation based on the QoS constraints mapping has been one of the difficulties faced by ISPs even with established TE mechanisms. Also, new technologies such as virtualization and cloud computing have enabled users to access their data and any content from anywhere and anytime. This has emphasized the need for efficient and fast resource allocation for providers to satisfy the user requests while improving their revenue [3].

Many researchers proposed multiple objective functions and resource allocation techniques to solve this network resource problem, which is discussed in detail in Section 2.6. One among such techniques is the network flow optimization problem where input is a flow network and an algorithm is designed to allocate appropriate weights on the links such that load acting on a link or utilization of a link is always less than its capacity. Most of the algorithms that have been proposed [4], [5], [6] and [7] considered the topology of the underlying network to be an undirected graph (see Section 2.2) and formulated the objective function and algorithm based on flow acting on each link of the network. The example of flow-based communication is the current flow between any two nodes which depends on all the resistances in a network. However, the data-communication network is generally a path-based network where transport of data packets between any two nodes is done through shortest paths in most of the cases and path-based properties are not considered in most of the cases while formulating the aforementioned objective function. Considering shortest paths alone may not give the complete overview of the flow in the network. Hence, there is a need to consider both flow-based as well as path-based network properties to arrive at an ideal network resource allocation solution which is possible through graph theory. Inverse shortest path algorithm (ISPA) is thus formulated considering both flow-based and path-based graph properties, considering the network to be an undirected graph with N nodes and L links proposed by P. Van Mieghem et al. [8] to address the network resource allocation problem in a holistic approach.

1.1. Challenge

The inverse shortest path algorithm is formulated on two bounds (see Section 3.5). It has been verified in [8] that inverse shortest path problem (ISPP) bounds hold for unweighted graphs and ISPA can be implemented for unweighted graphs. As most of the real-world networks are weighted graphs, it is necessary to extend the ISPA for weighted graphs as well. However, it is observed that these bounds do not always hold for weighted graphs, creating uncertainty for implementation of ISPA. The challenge of the thesis is therefore formed by the uncertainty of ISPA implementation for weighted graphs.

1.2. Objectives

The objectives of the thesis are therefore formulated as

1. To understand the nature of ISPA bounds for weighted random graphs with uniformly generated link weights through various metrics such as Q-Norm distribution, probability of failure distribution and hop count distributions.
2. To check the feasibility of ISPA implementation for weighted random graphs based on the relation between various parameters of the graph obtained by the analysis of the nature of ISPP bounds.
3. To implement ISPA for randomly generated demand matrices so as to obtain weighted adjacency matrices that are non-negative and to verify the feasibility of ISPA.

1.2.1. Approach

The following approach is followed in the thesis:

1. Perform literature study of network flow problems, resource allocation problems and various optimization techniques to understand the scope, limitations and latest developments in this field.
2. Based on ISPP lower bound (see Section 3.5), Q-Norm is computed and studied for various configurations of ER graphs to understand the dependency of each input parameter on the realization of ISPA for weighted graphs.
3. The Q-norm distributions and hopcount distributions are analysed for weighted ER graph for varying input parameters to determine the nature of ISPP bounds for weighted graphs.
4. The probability of failure of ISPP bounds (see Chapter 5) is studied for various configurations of ER graphs to derive the dependency of critical threshold r_0 of link resistances of the ER graphs on its input parameters such as number of nodes N, link density p and average link resistance r.
5. Finally, ISPA is implemented for a range of randomly demand matrices D to obtain non-negative solutions and the feasibility condition derived is verified for ISPA solutions.

1.2.2. Thesis Outline

The thesis is built-up as follows:

- Chapter 2 presents the literature study and related work that has been carried out in Network flow and resource allocation problems, various optimization techniques and also provides the latest developments as well as a comparison of various methods. It also gives the basics of graph theory which are necessary for understanding the Inverse Shortest path problem and its bounds.
- Chapter 3 explains the theory of ISPP and the theorems upon which ISPA is formulated and gives the need and scope of the thesis.
- Chapter 4 gives the Q-norm analysis of ISPP lower bound for weighted random graphs and its dependency on input parameters of ER graphs.
- Chapter 5 presents the probability of failure analysis for weighted graphs and the relation between various parameters of weighted graphs to derive a feasibility check for ISPA implementation.
- Chapter 6 describes the range of results possible with ISPA through hop count distribution analysis for various configurations of weighted ER graphs.
- Chapter 7 explains the validation of ISPA for a range of demand matrices and the verification of feasibility conditions of ISPP bounds.
- Finally, chapter 8 concludes the thesis summarizing the various results obtained in Chapters 4, 5, 6 and 7 and draws insights from the experiments carried out and gives recommendations for further research on this topic.

2 Background and related work

2.1. Introduction

This chapter gives an overview of various properties of graph metrics that have been utilised in understanding and formulating ISPA and ISPP. This chapter also gives the related study of network flow problems and resource allocation problems that have been studied to understand the scope of the project as well as to give the overview of the work that has been carried out in this field.

2.2. Graph

Arrangement of different points and their connection with each other in a layout formulates a graph which has been helpful in understanding the various underlying properties of the graph such as its degree distribution, betweenness of nodes or links of the graph, hopcount distribution, etc. This has led to the introduction of Graph Theory, a branch of discrete mathematics that is used in the study of various physical networks such as electric circuits, transport systems, brain networks, social networks, epidemics, etc. To understand the elegance of graph theory and its distinct and diverse properties, it is vital to understand the basic terminology and types of graphs.

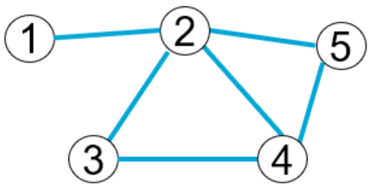


Figure 2.1: An undirected graph $G1(N, L)$ with number of nodes $N = 5$ and number of links $L = 6$ with unit link weights

A graph is given by $G(N, L)$ with N being the number of nodes (also called vertices) and L representing the number of links (also called edges) between different node pairs. It's generally assumed that a graph does not contain any self-loops (starting node and ending node of any link are not same) and multiple links between a node pair. For example, in the graph presented in Fig. 2.1, with set of nodes $n = \{1, 2, 3, 4, 5\}$ and set of links

$l = \{(1,2),(2,3),(2,4),(2,5),(3,4),(4,5)\}$, it can be observed that links in the graph $G1$ do not specify any direction i.e., the origination and destination of the link, hence these types of graphs are called undirected and no self-loops or multiple links between a node pair are present. In this project, all the graphs considered for analysis are undirected. Insights on another kind of graphs - directed graphs or digraphs and comparison of directed and undirected graphs and their applications in detail can be found in [9].

2.3. Random graph

A random graph is a graph whose edges occur in a certain probability and the occurrence of the links is random in nature. As the ISPP model input is demand matrix D (see Section 3.3), we may not be aware of the underlying topology of the graph. Hence, it is essential to consider the random graphs for the starting model of the ISPP study. In Subsection 2.3.1, we describe the two basic models for the random graphs.

2.3.1. Erdös Rényi Graphs

The first model on random graphs is put forth by Erdös Rényi, represented by $G(N,L)$ [10]. It states that if the maximum number of links in a graph is $L_{max} = \binom{N}{2}$, then the number of links in $G(N,L)$ is chosen uniformly with probability $\binom{L_{max}}{L}^{-1}$ and consists of $\binom{L_{max}}{L}$ elements.

The second model, the binomial model $G_p(N)$ proposed by Gilbert in 1959 [11] has number of nodes N and link probability given by $p = \dfrac{E[L]}{L_{max}}$ where $E[L]$ is the average number of links in graph G and $L_{max} = \binom{N}{2}$ is the maximum number of links possible. The degree distribution for this second model is binomial in nature and given by $Pr[D=k] = \binom{N-1}{k} p^k (1-p)^{N-1-k}$. The set of graphs obtained by this model lie in the extremes of $p = 0$ which gives a null graph and $p = 1$ which gives a complete graph. An undirected Erdös Rényi (ER) graph for number of nodes $N = 15$ and link probability $p = 0.25$ is presented in Fig. 2.2.

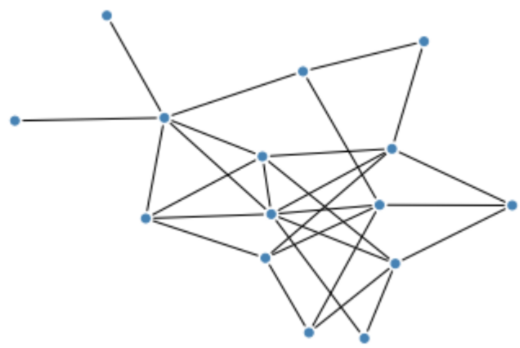


Figure 2.2: An undirected ER graph for number of nodes $N = 15$ and link probability $p = 0.25$

2.4. Adjacency matrix A

The adjacency matrix A gives the link existence between the nodes of the graph G in a $N \times N$ matrix, where each element of the matrix is either 0 or 1. If $a_{ij} = 1$, it represents the link existence between node pair (i,j) of graph G, else $a_{ij} = 0$. And $a_{ii} = 0$, as there are no self loops considered in our study. The adjacency matrix is symmetric hence $a_{ij} = a_{ji}$, in matrix form given as $A = A^T$. The adjacency matrix A for graph $G1(N, L)$ is therefore given as,

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

If the graph has diverse weights which are not unity, adjacency matrix which provides weight of the links along with the link presence is called weighted adjacency matrix represented by $\tilde{A}$. As an example, the graph $G2(N, L)$ is presented in Fig. 2.1 with non-uniform link weights and the corresponding weighted adjacency matrix $\tilde{A}$ is also given.

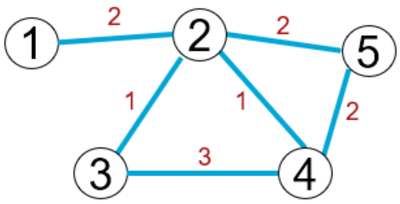


Figure 2.3: Graph $G2(N, L)$ with number of nodes $N = 5$, number of links $L = 6$ and non-uniform link weights.

$$\tilde{A} = \begin{bmatrix} 0 & 2 & 0 & 0 & 0 \\ 2 & 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 3 & 0 \\ 0 & 1 & 3 & 0 & 2 \\ 0 & 2 & 0 & 2 & 0 \end{bmatrix}$$

2.5. Graph Metrics

Graph metrics measure the properties of a graph based on the topology of the network such as connectivity, robustness, resilience, etc. In this Section, we will look into the graph metrics that have been utilised in the implementation of ISPA for weighted graphs. To clearly understand these graph related matrices and metrics, we will also derive the corresponding metrics for the graphs in Fig. 2.1 and Fig. 2.3 along with their definitions.

2.5.1. Degree

The degree of node i gives the number of nodes with which node i is connected by a link, i.e., the number of neighbors of node i. In a connected network, the degree of each node ranges from 0 to $N-1$. For an undirected graph $G(N, L)$, the sum of degrees of all the nodes is given by (2.1) and 2 in the equation occurs as the link between two nodes is counted twice.

$$\sum_{i=1}^{N} d_i = 2\ L \tag{2.1}$$

Hence the average degree of the graph G is given by (2.2).

$$d_{avg} = \frac{1}{N} \sum_{i=1}^{N} d_i = \frac{2L}{N} \tag{2.2}$$

The minimum value of average degree d_{avg} is $2-\frac{2}{N}$ obtained for spanning trees with number of links $L=N-1$. The maximum value for average degree d_{avg} is $N-1$ which is seen in the case of complete graphs with $L=\frac{N(N-1)}{2}$.

The degree vector $d=(d_1,d_2,....,d_N)$ is a column vector that gives the degree of each node of graph G in a vector obtained by the relation $d=A\,u$ where A is the Adjacency matrix of the graph and u is the all one vector. For the graph in Fig. 2.1,

- Sum of the degrees of all nodes $\sum_{i=1}^{N} d_i = 2\,L = 12$.
- Average degree of the graph $G1 = 2.4$
- Degree vector is thus obtained as

$$d = A*u = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 2 \\ 3 \\ 2 \end{bmatrix}$$

2.5.2. Shortest path matrix S

In a weighted graph, each link in graph $G(N,L)$ is assigned a weight given by w_l for link $l\epsilon\ L$ which represents any property of the network such as resistance, delay, cost incurred, capacity, etc. Weight of the path P_{ij} between node pair (i,j) is given by the sum of the weights of the links along the path taken.

$$w(P_{ij}) = \sum_{l\epsilon P_{ij}} w_l \quad (2.3)$$

The shortest path $P^*_{i\to j}$ from node i to node j is the path with minimum weight. Hence,

$$w(P^*_{i\to j}) \le w(P_{i\to j}) \text{ for all } P_{i\to j} \quad (2.4)$$

The shortest path matrix S consists of weights of the shortest paths between all the node pairs of the graph. In an unweighted graph, the shortest paths between node pairs are also called as shortest hop paths. For the graph $G1$ in Fig. 2.1, the S matrix is computed as $S1$ and for the weighted graph $G2$ in Fig. 2.3, the S matrix is computed as $S2$

$$S1 = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 1 & 1 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 2 & 1 & 0 \end{bmatrix} \qquad S2 = \begin{bmatrix} 0 & 2 & 3 & 3 & 4 \\ 2 & 0 & 1 & 1 & 2 \\ 3 & 1 & 0 & 2 & 3 \\ 3 & 1 & 2 & 0 & 2 \\ 4 & 2 & 3 & 2 & 0 \end{bmatrix}$$

The number of links or hops in the shortest path between nodes i and j is called the hopcount between node pair (i,j) generally represented by h_{ij}. The hopcount between all the node pairs in a graph $G(N,L)$ is given in a distance matrix H. It's called distance matrix as it obeys distance relations such as

- Each element of distance matrix $h_{ij} \ge 0$.
- Diagonal elements are always zero: $h_{ii} = 0$.
- It obeys triangular inequality: $h_{ik} + h_{kj} \ge h_{ij}$

The distance matrix of the graph $G1(N, L)$ is calculated as $H1$ and the distance matrix of $G2(N, L)$ is computed as $H2$. In $G2(N, L)$, due to link weights w_{32} and w_{24} influence, the shortest path between $w(P^*_{3\to 4})$ is routed via $3 \to 2 \to 4$ instead of direct path $3 \to 4$.

$$H1 = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 1 & 1 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 2 & 1 & 0 \end{bmatrix} \qquad H2 = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 1 & 1 \\ 2 & 1 & 0 & 2 & 2 \\ 2 & 1 & 2 & 0 & 1 \\ 2 & 1 & 2 & 1 & 0 \end{bmatrix}$$

2.5.3. Effective resistance matrix Ω

It gives the effective resistance between all the node pairs of the graph in a matrix. For an electrical network, the effective resistance between the end points of the network is computed by series and parallel combinations of resistances. In graph theory, we compute effective resistance matrix with the use of Laplacian matrix Q given by

$$Q = \Delta - A \text{ where } \Delta = \text{ Degree matrix} = \begin{bmatrix} d_1 & 0 & ... & 0 \\ 0 & d_2 & ... & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & \dots & d_n \end{bmatrix} \tag{2.5}$$

and $d_i =$ degree of node i and $A =$ Adjacency matrix of graph G.

From Q, the pseudo inverse $Q^\dagger$ is computed as

$$Q^\dagger = \sum_{k=1}^{N-1} \frac{1}{\mu_k} z_k \, z_k^T \tag{2.6}$$

where $\mu_k = k^{th}$ eigen value when all the eigen values of Q are arranged in $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_{N-1} \geq \mu_N \geq 0$ and $z_k =$ eigen vector corresponding to μ_k. The effective resistance matrix is given by

$$\Omega = u\zeta^T \; + \; \zeta u^T \; - \; 2Q^\dagger \tag{2.7}$$

where $\zeta =$ Diagonal vector of $Q^\dagger = (Q^\dagger_{11}, Q^\dagger_{22}, \dots, Q^\dagger_{NN})$.

The effective graph resistance R_G is a graph property by which the difficulty of transport is measured in graph G [12] given by

$$R_G = \frac{1}{2} \sum_{a=1}^{N} \sum_{b=1}^{N} \omega_{ab} = \frac{1}{2} u^T \Omega u \tag{2.8}$$

where $\omega_{ab} =$ Effective resistance between node pair (a, b).

For weighted graph, weighted effective resistance matrix $\tilde{\Omega}$ and weighted effective graph resistance $\tilde{R_G}$ are computed in similar manner with inputs as flow-based weighted adjacency matrix $\tilde{A_F}$ and the corresponding weighted laplacian matrix $\tilde{Q} = \tilde{\Delta_F} - \tilde{A_F}$.The flow-based weighted adjacency $\tilde{A_F}$ is constructed with elements af_{ij} where $af_{ij} = \dfrac{1}{a_{ij}}$ for all $i \neq j$,

$af_{ij} = 0$ for all $i = j$ and if there is no link present between node pair (i, j) i.e. if $a_{ij} = 0$ then $af_{ij} = 0$ and $\tilde{\Delta_F}$ = Degree matrix of A_F.

For unweighted graph $G1$ in Fig. 2.1, the effective resistance and effective graph resistance are computed as

$$\Omega = \begin{bmatrix} 0 & 1 & 1.625 & 1.5 & 1.625 \\ 1 & 0 & 0.625 & 0.5 & 0.625 \\ 1.625 & 0.625 & 0 & 0.625 & 1 \\ 1.5 & 0.5 & 0.625 & 0 & 0.625 \\ 1.625 & 0.625 & 1 & 0.625 & 0 \end{bmatrix} \qquad R_G = 9.75$$

For weighted graph $G2$ in Fig. 2.3, the weighted effective resistance and weighted effective graph resistance are computed as

$$\tilde{\Omega} = \begin{bmatrix} 0 & 2 & 2.792 & 2.667 & 3.167 \\ 2 & 0 & 0.792 & 0.667 & 1.167 \\ 2.792 & 0.792 & 0 & 1.125 & 1.792 \\ 2.667 & 0.667 & 1.125 & 0 & 1.167 \\ 3.167 & 1.667 & 1.792 & 1.167 & 0 \end{bmatrix} \qquad \tilde{R_G} = 17.334$$

2.6. Network resource allocation problem

The network resource allocation problem is the shortest path routing problem where the shortest path between any two vertices is calculated, given the resources consumed along a path must lie with the demand acting on the link. These network resource allocation problems have been formulated in various domains such as transportation, telecommunications etc. [13]. In this Section, we will see the various network resource allocation problems utilised in Traffic Engineering (TE) to satisfy the network resource accommodation such as bandwidth, Quality of Service (QoS) delay, etc. and various objective functions formulated to solve the routing problem and arrival on General Routing Problem. The Section also discusses the various methods by which the routing optimization was addressed by numerous techniques and discusses the benefits and drawbacks of these techniques.

The intra-domain routing problem was mostly presented as minimization of the maximum utilization of the network in many of the routing problems [14], [15] and [5]. Consider a directed graph $G(N, L)$ where N represents the nodes in the Graph and L is the set of links between the nodes in the graph. The utilization of the link (i, j) is defined as $U_{ij} = l_{ij}/c_{ij}$ where l_{ij} is the load acting on the link $(i, j) \in L$ and c_{ij} is the capacity of the link $(i, j) \in L$. The objective function was thus formulated as (2.9).

$$\text{min max}_{(i,j)\,\epsilon\,L} U_{ij} \tag{2.9}$$

As the objective function has been aimed at reducing the demand of major utilised link, it has not been global to the network, i.e., the whole network was not considered for optimization.

The major work in shortest path routing optimization was performed by Fortz and Thorup by developing a local heuristic approach to solve the cost optimization problem. They proposed a piece-wise linear cost function in which each link is assigned a routing cost based on an increasing convex optimization function in the work of [4], [16] and [17]. The cost function $\Phi(i, j)(c_{ij}, l_{ij})$ has been modelled based on the closeness of l_{ij} to its capacity c_{ij}. The objective function is formulated as to keep the load within the capacity of each link while minimizing the overall cost of the network. The change in cost for increased or decreased load acting on the link is presented as derivative of Φ as follows in [16].

$$\phi'(c_{ij}, l_{ij}) = \begin{cases} 1 & \text{for} \quad 0 \leq l_{ij}/c_{ij} \leq 1/3 \\ 3 & \text{for} \quad 1/3 \leq l_{ij}/c_{ij} \leq 2/3 \\ 10 & \text{for} \quad 2/3 \leq l_{ij}/c_{ij} \leq 9/10 \\ 70 & \text{for} \quad 9/10 \leq l_{ij}/c_{ij} \leq 1 \\ 500 & \text{for} \quad 1 \leq l_{ij}/c_{ij} \leq 11/10 \\ 5000 & \text{for} \quad 11/10 \leq l_{ij}/c_{ij} \leq \infty \end{cases} \tag{2.10}$$

The cost function Φ has been modelled as an increasing piece-wise linear convex function symbolising that links which are over-utilised are penalised for allowing to transmit traffic when the capacity is fully utilised. If the utilisation is higher than 100% and till 110% cost of the link increases rapidly. The cost calculation with respect to load acting on the network is presented in Fig. 2.4 where capacity of the link is considered to be 1 and the cost function is modelled as increasing function of load on the link.

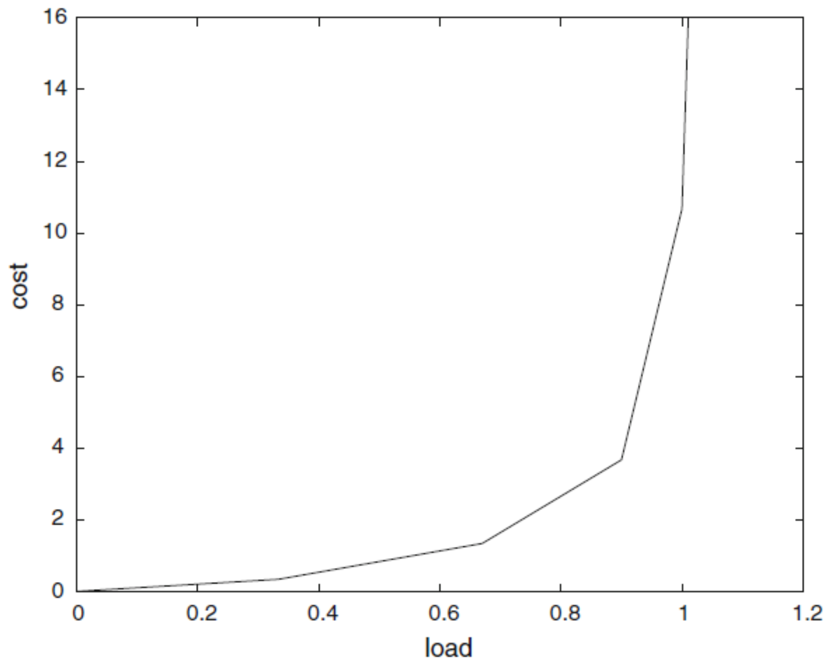


Figure 2.4: Link cost $\Phi(1, l_{ij})$ versus load acting on the link when capacity is unity.

This model is the first one to address the link weight optimization based on the utilisation. Their work acted as an impetus for many researchers to solve the optimal routing problem. The work of Fortz and Thorup [4] and [17] also formulated the General Routing Problem (GRP) as minimization of the objective function Φ with respect to the demand acting on each link. GRP has been expressed as a Multi Commodity Network Flow Problem (MCNFP) - the set of demands are routed across the links based on the link capacities. However, this GRP doesn't take the Equal Cost Multi-Path Routing principle (ECMP) into consideration which makes the GRP as the lower bound for Interior Gateway Routing Protocol (IGP). But Fortz et al. [4] proved that including ECMP rule would make IGP routing NP hard. This NP-completeness has also been presented in the work of [18] and [15].

As a workaround to circumvent the NP-completeness problem, a local search heuristic approach was proposed in the work of Fortz and Thorup [4] and [17] to optimize the GRP. In this approach, the function to be minimised is supposed to be g over a set of Y possible option sets. For each iteration of the algorithm, a neighbourhood $S(y) \subset Y$ was chosen to minimise the function g such that $g(y)$ bares lesser cost than the previous neighbourhood. The techniques such as search diversification and hashing were utilised for faster solutions [4]. The method was further improved by additional weight settings for the possibility to include link failures and robustness in [16].

The work of Kodialam and Lakshman [19] introduced a new objective function called minimum interference routing to address the network resource allocation problem. The objective function is given as:

$$\max \sum_{(i,j)\epsilon Q} \alpha_{st}\theta_{st} \tag{2.11}$$

where α_{st} is the weight representing the importance of ingress-egress pair $(s,t)\ \epsilon\ Q-$ set of all commodities. The motive of the objective function is to minimize the blocking probability of the future demand request by minimising the maximum flow between a pair $(s,t)\ \epsilon\ Q$ such that the maximum future request that can be admitted between the pair also reduces. Their work aimed to maximize the weighted sum of the maximum flows over residual topology. Similar works of residual capacity maximization have been discussed in [15] and [5]. But, the objective function fails to incorporate multiple criteria into account. As network performance is dependent on various criteria such as throughput, fairness, utilization, etc. the works of [4],[16],[17] and [19] didn't provide the holistic optimal solution to address the intra domain routing problem.

Degrande et al. [20] put forth a weighted sum of multiple criteria method by formulating an objective function incorporating various bases by introducing coefficient for each parameter to give priority to different criteria based on the need of the network optimization. The objective function was, thus formulated as

$$\min(C_b\, U^{max} + C_u \sum_{(i,j)\epsilon L} U_{ij}) \tag{2.12}$$

where C_b is the coefficient associated with balance $B = 1 - U^{max}$ and C_u is the coefficient associated with the network utilization defined as $U = \sum_{(i,j)\epsilon L} U_{ij}$.

The work of Pioro and Medhi [15] and [5] have also contributed to shortest path routing optimization. Their work was mainly focused on residual capacity optimization and minimum average delay Objective Functions. Pioro and Medhi [15] and [5] formulated the generalised Interior Gateway Protocol (IGP) weight optimization problem as follows:

$$(IGP_{basic}) \quad \{Objective\} \tag{2.13}$$

$$\text{such that} \sum_{p\epsilon P} f^p_{st}(w) = d_{st}(s,t)\epsilon Q$$

$$\sum_{p\epsilon P} \sum_{(s,t)\epsilon Q} \delta^{st}_{pa} f^p_{st}(w) \leq \gamma_a c_a$$

where IGP_{basic} is the Intra Domain Routing Objective Function with any objective which meets the network requirements, $P(s,t)$ represents the set of directed paths for the origin-destination pair $(s,t)\ \epsilon\ Q\ -$ set of commodities. f^p_{st} depicts the traffic flow due to the weight

ω acting on path $p \epsilon P(s,t)$. d_{st} represents the traffic demand for the pair $(s,t)\epsilon Q$. δ_{pa}^{st} is 1 if link a belongs to path $p\epsilon P(s,t)$ and 0 otherwise. γ_a shows the link utilization factor for link a belongs to path $p\epsilon P(s,t)$. The IGP was later modified to meet the ECMP requirement as well. The equations (2.13) - (2.6) thus served as the basis for generalised routing function for intra domain routing problem. Their work also provides the NP-completeness to optimize IGP metric with ECMP rule.

Pioro and Medhi have also proposed Weight Adjustment method in [5] to solve the routing objective. In this method, the weights of the links are initially set to a random number and adjusted till a preferred load optimisation is achieved. The weight variation method is carried out such that the least and most loaded links are chosen and the weights are modified until the required network utilization is achieved. Though this method optimizes the network utilization, the iterative nature of the algorithm may lead to recursive action there by consuming the computational resources. They have also proposed a Lagrangean Relaxation approach which utilises the objective functions of residual capacity maximization to obtain the optimal link weights.

The various objective functions were compared in the work of [21] for the best optimal routing method. The objective functions were tested on an operational network and the United States research network called Abilene with diverse traffic matrices. It has been verified that the objective function of [4] performs better compared to the rest of the objective functions.

The Intra-domain routing problem was also addressed by other heuristic algorithms such as Genetic Algorithm [7]. This algorithm utilised cross-over procedure where two parents $p1$ and $p2$ are combined to produce a random vector of real numbers. A cut-off number $V \epsilon [0.5, 1]$ was chosen to determine the gene of the child. This approach was further improved by adding local search algorithm in [22]. The resulting approach has led to faster convergence compared to that of [4].

The methods so far discussed the routing problem incorporating ECMP principle. Bley has proposed unique shortest path problem - link weights are chosen such that demand acting on the network are routed by unique shortest paths in [6]. This approach is also referred as inverse shortest path problem. In this work [6], unique shortest path problem was formulated as Integer Linear Programming models to minimize the longest arc or longest path length.

The major works of network resource allocation problem in Traffic Engineering domain are presented in Table 2.1 which gives an outlook of the optimization objective chosen, optimization method developed and the outcome of the work. The common thread in all these works is that the demand acting on the network is modelled as network flow problem. The works mentioned in Table 2.1, give the enormous possibilities of optimization methods possible for solving the network resource allocation problem. In this project, we have formulated a novel algorithm based on graph theory and its metrics.

Reference	Optimization Objective	Optimization Method	Performance/Outcome
Fortz and Thorup [4], [16], [17]	Optimizing weight settings based on a demand matrix subject to minimize the overall cost	Local Search Heuristic	50-110 % increase in demand is met
Pioro et al. 2000 [15]	Maximization of average free capacity and total free capacity	Two phased approach - Mixed Linear Integer programming (Phase 1), linear programming (Phase 2)	Optimal
Altin 2013 [14]	Minimization of maximum utilization	Mixed Integer Linear Programming - Branch And Price Algorithm	Optimal
Bley 2005 [6]	Minimization of longest path/arc	-	Proof of NP-Completeness of Inverse Shortest path problem
Balon 2006 [21]	Comparison of various objective functions such as weighted mean delay, multiple criteria	Linear Programming model and simulations	Delay objective function is optimal and best objective function
Wang et al. 2001 [23]	Minimize the total weight of links - Dual Shortest path formulation	Linear Programming Duality	Optimal
Blanchy et al. 2003 [24]	Load balancing function	Bellman- Kalaba Algorithm	Optimal
Elwalid 2001 [25]	Average delay function	MATE - Asynchronous Algorithm	Optimal
Degrande 2003 [20]	Minimize blocking, utilization rate; Maximizing throughput and load balancing	Mixed Integer Linear Program	Less Optimal for flat networks
Holmberg and Yuan 2004 [26]	Minimization of cost	Lagrangian Heuristic Approach and branch bound Algorithm	Optimal
Ericsson et al. 2002 [7]	Minimization of network congestion	Genetic Algorithm	Optimal
Farago et al. 2003 [27]	Minimization of cost vector	Combinatorial Optimization and linear programming	Reduction in average blocking probability from 20 % to 3.5 %
Bley and Koch 2002 [6]	Minimization of flow and installation cost for access and backbone network	Mixed integer linear programming model for each access and backbone network	Optimal
Kodialam and Lakshman 2000 [19]	Minimum Interference Routing	Minimum Interference Routing Algorithm (MIRA)	Optimal
Zhang and Rodosek 2005 [28]	Maximize the sum of residual capacities	Mixed Linear Programming with Bender's Decomposition Method	Optimal

Table 2.1: Various research objective functions and methods proposed by peers along with the performance/outcome of the research carried out.

2.7. Conclusion

This chapter gives the basics of graph theory, random graph models and graph metrics needed to understand the thesis. The graph metrics are computed for example graphs to understand the basics in an effective way. Alongside, network resource allocation problem and various works by peers in this domain have been studied to understand the scope and novelty of the inverse shortest path algorithm.

3

Methodology: Inverse Shortest Path Problem

3.1. Introduction

We have seen the various graph models, network resource allocation problems, various objective functions and algorithms formulated by peers to solve the network resource allocation problem in the previous chapter. In the current chapter, we will be looking into the inverse shortest path problem (ISPP), inverse shortest path algorithm (ISPA) along with its bounds. The aim of this chapter is to provide the methodology upon which ISPA is formulated and to understand its applicability for weighted graphs and the need of the thesis. The Sections 3.4, 3.5, 3.6 and 3.7 are incorporated from [8] which have been formulated by P. Van Mieghem et al. and the article can be referred for in-detailed description, as the theory in this chapter is bounded to provide the scope of the thesis.

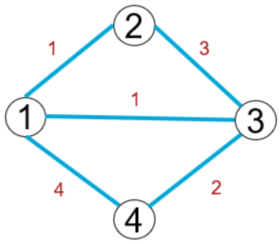
3.2. Flow and path networks

ISPP is a network resource allocation problem formulated based on flow and path network-based graph metrics and properties. To understand the ISPP, it is therefore vital to understand the flow and path networks.

Consider a graph G with a set of N nodes and L links. Let w_l be the weight on link $l \in L$, which gives any property of the network such as delay, bandwidth, throughput etc. For example, an electrical grid network has link weights as resistances in ohm whereas the transportation network has link weights being the distance between various nodes.

Firstly, let us consider the electrical grid which is the best example of flow-based communication. Fig. 3.1 is an electrical network represented in the form of undirected graph $G(N, L)$ where the link weight represents the resistance between any two node pairs in the network. The effective resistance between any node pair, say, $(1, 2)$ given by R_{12} is not only dependent on the flow between node pair $(1, 2)$ but also on all the flows in the network. The effective resistance matrix Ω is the graph metric which gives the effective resistance between all node pairs in the Graph and is discussed in Section 2.5.3.

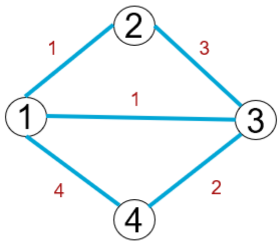
On the other hand, a path network is a network that is utilised in data communication to transfer digital information between various nodes and for this type of communication, the



$$\Omega = \begin{bmatrix} 0 & 0.794 & 0.705 & 1.647 \\ 0.794 & 0 & 1.147 & 2.206 \\ 0.705 & 1.147 & 0 & 1.412 \\ 1.647 & 2.206 & 1.412 & 0 \end{bmatrix}$$

Figure 3.1: A simple electrical grid network represented in an undirected graph $G(N, L)$ along with its effective resistance matrix Ω

path with the shortest length is chosen to transfer the data. The weight of the path $w(P_{ij}) = \sum_{l \,\epsilon\, P_{ij}} w_l$ of a path P_{ij} between node pair (i, j) consists of the sum of the weights over all the links that belong to path P_{ij}. The shortest path P^*_{ij} is the path with minimum weight of all the paths P_{ij} and hence $w(P^*_{ij}) \leq w(P_{ij})$. The shortest path matrix S is the graph metric which gives the shortest path between all the node pairs of the graph G as shown in Fig. 3.2.



$$S = \begin{bmatrix} 0 & 1 & 1 & 3 \\ 1 & 0 & 2 & 4 \\ 1 & 2 & 0 & 2 \\ 3 & 4 & 2 & 0 \end{bmatrix}$$

Figure 3.2: An undirected graph $G(N, L)$ along with its shortest path matrix S

These two graph metrics Ω and S are necessary to understand the ISPP bounds – bounds upon which ISPA is formulated which is discussed in detail in Section 3.5.

3.3. Network Communication Constraint

The network communication constraint, also called demand matrix D, gives the requirement of the network in terms of Bandwidth, QoS, Delay or cost etc. for each node pair (i, j) given by d_{ij}. Hence the demand matrix is given by,

$$D = \begin{bmatrix} 0 & \dots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \dots & 0 \end{bmatrix}$$

The demand matrix D is mainly used in resource constrained shortest path problem where shortest path between two vertices in the network is calculated, given the resources consumed along a path must lie below certain upper limit. This is often referred to as traveller problem with certain budget who has to reach a certain destination within the constraints imposed by the budget [29].

3.4. Inverse shortest path problem

The Inverse Shortest Path Problem (ISPP) is a form of resource constrained shortest path problem famously known as network resource allocation problem given by:

Given a $N \times N$ demand matrix D, determine the weighted Adjacency matrix $\tilde{A}$ such that obtained shortest path matrix S obeys

$$s_{ij} \leq d_{ij} \text{ for all } i = 1, 2, ..., N$$
$$\text{for all } j = 1, 2, ..., N$$

The above ISPP problem is solved using an inverse shortest path algorithm (ISPA) which utilises the ISPP bounds (Section 3.5) and Fiedler's inverse block matrix relation (Section 3.6) which are formulated using the graph metrics Ω and S.

3.5. ISPP Bounds

The ISPP bounds are the basis on which the ISPA is formulated. It essentially has two bounds to establish relation between two parameters Ω and S. The first bound states that weight of the shortest path $w(P^*_{ij})$ is lower bounded by the effective resistance for node pair (i, j). Mathematically written as,

$$\omega_{ij} \leq s_{ij} = w(P^*_{ij}) \tag{3.1}$$

If we construct a difference matrix $C = S - \Omega$, then the only case when $C = 0$ is observed for path graphs such as line graph, tree graph and star graph in which Ω and S coincide. In any other type of graph, the effective resistance Ω is always less than S which implies $C \geq 0$ due to which it results in parallel combination of resistances between the node pairs of the graph reducing the effective resistance.

The second bound upon which ISPA is formulated as 'the effective resistance ω_{ij} can be lower bounded by the combinatorial shortest path $\hat{s_{ij}}$' as,

$$\frac{1}{m}\hat{s_{ij}}^2 \leq \omega_{ij} \tag{3.2}$$

where $m = \sum_{l \,\epsilon\, L} w_l^{-1}$ is the sum over all links of the inverse link weights and $\hat{s_{ij}}$ is the weight of the shortest path $w(P^*_{ij})$ when each link has unit weight $w_l = 1$ for all $l \,\epsilon\, L$ and called hop count [12]. The second bound in (3.2) is based on theorem of Nash Williams' inequality [30] and the theorem is as follows:

Theorem: A cut in graph theory is described as a partition of the vertices of a graph into two disjoint sets. For a pair of nodes i and j, an $i - j$ cut consists of a set of links such that removing these links from the graph disconnects node i from node j. If C_{ij} is a collection of $i - j$ cuts which are independent i.e., no two cuts share a link, then Nash William's inequality states that

$$\sum_{C \,\epsilon\, C_{ij}} w(C) \leq \omega_{ij} \tag{3.3}$$

where $w(C) = \left(\sum_{(a,b)\epsilon\, C} w_{ab}^{-1}\right)^{-1}$ is the weight of a cut $C \,\epsilon\, C_{ij}$ and ω_{ij} is the effective resistance between two nodes $i \,\epsilon\, N$ and $j \,\epsilon\, N$.

For nodes i and j which are $\hat{s_{ij}}$ hops removed from each other, we consider the following collection of $i-j$ cuts $C_{ij} = \{C_k\}_{k=0}^{\hat{s_{ij}}-1}$, where the cut $C_k = \{(a,b) \,\epsilon\, L \,:\, \hat{s_{ia}} = k, \hat{s_{ib}} = k+1\}$ contains all links between one node at combinatorial shortest path distance k from i, and the other node at distance $k+1$. From (3.3), we obtain the lower bound as,

$$\omega_{ij} \geq \sum_{k=0}^{\hat{s_{ij}}-1} w(C_k) \tag{3.4}$$

By multiplying both sides of (3.3) with $m = \sum_{l \,\epsilon\, L} w_l^{-1}$, we obtain (3.5) that proves the lower bound in (3.2).

$$m\omega_{ij} \geq \sum_{k=0}^{\hat{s_{ij}}-1} w^{-1}(C_k) \geq \hat{s_{ij}}^2 \tag{3.5}$$

3.6. Fiedler's inverse block relation

In undirected flow networks, Fielder has derived an inverse block matrix relation [31], [32] from which the weighted adjacency matrix $\tilde{A}$ can be derived from effective resistance matrix Ω [33], [34], [35] as follows:

$$\begin{bmatrix} 0 & u^T \\ u & \Omega \end{bmatrix}^{-1} = \begin{bmatrix} -2\sigma^2 & p^T \\ p & -\frac{1}{2}\tilde{Q} \end{bmatrix} \text{ with } \Omega p = 2\sigma^2 u \tag{3.6}$$

where $\tilde{Q} = \tilde{\Delta}_F - \tilde{A}_F$ is the weighted Laplacian matrix and the variance $\sigma^2 = \frac{\zeta^T \tilde{Q} \zeta}{4} + R_G$, where $R_G = \frac{1}{2} u^T \Omega u$ is the effective graph resistance. The matrix $\tilde{\Delta}_F = diag(\tilde{A}_F u)$ is a diagonal matrix and u is the all-one vector. The diagonal elements of pseudo-inverse $Q^\dagger$ of the Laplacian $\tilde{Q}$ is ζ. The weighted degree vector $\tilde{d} = \tilde{A}_F u$ has components equal to diagonal elements of $\tilde{Q}$. The matrix $\tilde{A}_F$ represents adjacency matrix of the flow network and is different from the weighted adjacency matrix $\tilde{A}$ of the path network. In order to make weight of the link $w_l = r_l$ resistance of the link, the weighted Laplacian $\tilde{Q}$ elements q_{ij} can be made $-\frac{1}{r_{ij}}$ for $i \neq j$ [33]. Using this relation, $(\tilde{a}_F)_{ij} = \frac{1}{r_{ij}}$ while $\tilde{a_{ij}} = r_{ij}$, $(\tilde{a}_F)_{ii} = \tilde{a}_{ii} = 0$ and $(\tilde{a}_F)_{ij} = 0$ while $\tilde{a}_{ij} = 0$. By applying the block inverse formulae to Fielder's block matrix equation in 3.6, we obtain $2\sigma^2 = \frac{1}{u^T \Omega^{-1} u}$ and $p = \frac{1}{u^T \Omega^{-1} u} \Omega^{-1} u$ and thus the inverse of effective resistance matrix is obtained as

$$\Omega^{-1} = \frac{1}{2\sigma^2} p.p^T - \frac{1}{2}\tilde{Q} \tag{3.7}$$

Hence, the weighted adjacency matrix from (3.7) as

$$\tilde{A}_F = \tilde{\Delta}_F + 2\Omega^{-1} - \frac{1}{\sigma^2} p.p^T \tag{3.8}$$

By suitably converting the demand matrix D into a distance matrix, we can replace Ω by D in (3.8) and derive the weighted adjacency matrix $\tilde{A}_F$ which is the key principle of inverse shortest path algorithm (ISPA).

3.7. Inverse shortest path algorithm (ISPA)

By combining the two bounds in Section 3.5, we obtain the inequality in (3.9).

$$\frac{1}{L}s_{ij}^2 \leq \omega_{ij} \leq s_{ij} \tag{3.9}$$

As the number of links $L \leq \dfrac{N(N-1)}{2} < \dfrac{N^2}{2}$, the term L in lower bound can be replaced by $\dfrac{N^2}{2}$. Hence, the second bound in (3.2) can be written as $\dfrac{2}{N^2}s_{ij}^2 \leq \omega_{ij}$. By modifying to maintain S on the left hand side of the inequality, we arrive at $s_{ij} \leq \dfrac{N}{\sqrt{2}}\sqrt{\omega}_{ij}$. By choosing $\omega_{ij} = \dfrac{2}{N^2}d_{ij}^2$, in matrix form $\Omega = \dfrac{2}{N^2}D \ o \ D$, then the inequality in (3.9) can be written it terms of demand matrix D as,

$$\frac{2}{N^2}s_{ij}^2 \leq \frac{2}{N^2}d_{ij}^2 \tag{3.10}$$

Therefore, by choosing $\omega_{ij} = \dfrac{2}{N^2}d_{ij}^2$, we establish $s_{ij} \leq d_{ij}$ which in turn states that length of the shortest path is within the demand constraint. This is the first step in ISPA which requires to obtain Ω from the demand matrix D. The ISPA is formulated as follows:

Algorithm 1: Inverse shortest path algorithm

1. Choose $\Omega \leftarrow \dfrac{2}{N^2}(D \ o \ D)$
2. If ($\Omega \neq$ distance matrix of a simplex)
3. Construct a distance matrix $\Omega' \leq \Omega$
4. Construct a simplex distance matrix $\Omega'' \leq \Omega'$
5. Replace $\Omega \leftarrow \Omega''$
6. Compute $\tilde{A}_F$ via (3.8)
7. Return $\tilde{A}$ by $a_{ij} = \dfrac{1}{(A_F)_{ij}}$ for all $i \neq j$ and $(A_F)_{ij} \neq 0$

The demand matrix *D* may not be always a distance matrix. However, after obtaining effective resistance matrix Ω from D, it is possible to convert Ω to a distance matrix if it is not already one. After constructing Ω from line 1 of the algorithm, if $\omega_{ij} < 0$, then ISPP is not feasible. If Ω is not symmetrical, i.e., $\omega_{ij} \neq \omega_{ji}$, then ω'_{ij} is computed as $\omega'_{ij} = min(\omega_{ij}, \omega_{ji})$. If $\omega_{ik} + \omega_{kj} < \omega_{ij}$ for at least one $k \epsilon N$ which violates the triangular inequality of a distance matrix, then we replace $\omega'_{ij} = min_{1 \leq k \leq N} \ (\omega_{ik} + \omega_{kj})$. In line 4, we check whether Ω corresponds to a simplex, else the calculated $\tilde{A}_F$ has negative elements. However, it is not possible to check the simplex nature of Ω as it is complicated and can be taken as future study.

3.8. Need of the thesis

The ISPA is possible to implement for unweighted graphs as the two bounds (3.1) and (3.2) hold as link weights are all unity. In case of a weighted graphs, it has been observed that lower bound (3.2) doesn't hold always and the variability of the inequality in (3.2) changes with input parameters of the graph such as number of nodes N, link density p and average link resistance r for ER graphs. Hence the need of the thesis is to validate the implementation of ISPP for weighted graphs and also to study its characteristics through probability of failure, Q-Norm analysis and hop count distribution analysis to determine the conditions under which ISPP lower bound holds for weighted graphs and to understand the nature of the solutions obtained by applying ISPP for weighted graphs.

4

Q-norm Analysis

4.1. Introduction

In the previous Chapter Methodology, the ISPP bounds upon which ISPA works have been studied in detail and we also observed that ISPP upper bound (3.1) holds for weighted graphs and unweighted graphs. Also, the ISPP lower bound (3.2) behaviour is unknown for weighted graphs and will be explored through the Euclidean norm or Q-norm analysis carried out in this chapter. Through Euclidean norm analysis, we aim at understanding the ISPP lower bound behaviour in weighted graphs and its dependency on the configuration parameters such as number of Nodes N, link density p and average link resistance r for ER graphs qualitatively.

Euclidean norm is a function on real or complex vector space, where Euclidean distance of a vector is computed from the origin. If $x = (x_1, x_2, ...x_n)$ is a vector in the n-dimensional Euclidean space R^n , Euclidean norm or Q-norm [36] is given by $\|x\|_2 = \sqrt{x_1^2 + x_2^2 + ... + x_n^2}$. In ISPP, Q-norm is computed for the ISPP lower bound to find the closeness between the terms Ω and $(1/m) * S\ o\ S$ and to determine if the variation between these terms is dependent on the input or configuration parameters of the graph such as number of nodes N, link density p and average link resistance r for weighted ER graphs. The procedure followed is explained in Section 4.2.

4.2. Simulation setup

The experiment is carried out for weighted ER graphs for the following input parameters.

- Number of nodes $N = \{25, 50, 75, 100\}$
- Link density $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$
- Average link resistance r ranging starting from 0.1 with incremental of 0.5.
- Minimum link resistance $r_{min} = 0.1$.
- Maximum link resistance $r_{max} = 2 * r - 0.1$.
- Number of realizations for each N, p and r of ER graphs $= 10^6$.

For each N, p and r, 10^6 instances of weighted ER graphs are generated with uniformly distributed weights. Initially, the topology or the underlying structure of the graph is generated for a given N and p with unit link weights and for each ER graph structure, uniformly distributed link weights in the range of r_{min} and $r_{max} = 2 * r - r_{min}$ are generated and assigned to the underlying structure to make it weighted. For each weighted ER graph constructed, Effective resistance matrix Ω (see Section 2.5.3), Shortest path matrix S are computed (see Section 2.5.2) along with m which is sum of inverse link weights of the graph. For each weighted ER graph generated, the Q-norm is computed as

$$\left\| \Omega - \frac{1}{m} * S \ o \ S \right\|_{q=2} \text{ if } \Omega \geq \frac{1}{m} * S \ o \ S \tag{4.1}$$

Thus, for 10^6 realizations of weighted ER graphs generated for a combination of N, p and r, the total number of Q-norm values can be anywhere between a maximum of 10^6 when the inequality holds for every instance to a minimum of 0 when no weighted ER graph instance satisfies the inequality. The matlab code for generating Q-norm distributions is presented in Appendix A.

To understand the variation of each of the input parameters N, p and r on the ISPP bounds, Q-norm distribution analysis has been performed where the set of Q-norm values obtained for a combination of N, p and r of ER graph are normalized by the total number of Q-norm values and the normalized Q-norm values are plotted for variation in one of the input parameters keeping the other two parameters constant. So, the Q-norm distribution analysis is studied for the following combinations and the insights derived are discussed in Section 4.3.

1. Variation in N for constant p, r
2. Variation in p for constant N, p
3. Variation in r for constant N, p

4.3. Analysis

The Q-norm distribution analysis plotted for variation in each input parameter for the other parameters being constant is shown in Fig. 4.1, Fig. 4.2 and Fig. 4.3 and the following insights are derived.

- The Q-norm distributions follow a binomial distribution for $N \geq 50$ with corresponding mean μ and standard deviation σ given by $Pr[X = k] = \binom{n}{k} p^k (1-p)^{n-k}$ where X is a binomial random variable, where p being probability of success on a single trial and n being the total number of trials. For lower N and lower p, it is observed that the distributions distort which may be due to their finiteness and are deviated from the binomial distribution.
- As number of Nodes N increases for a constant p and r in Fig. 4.1, the mean of the distribution moves closer to origin indicating the closeness of bounds. This means the ISPP bounds are valid for higher N $(N >= 50)$ with higher probability.
- As link density p increases for constant N and r in Fig. 4.2, the mean of the distribution approaches origin indicating the difference between the terms in inequality reduces. This could be due to increase in number of links within the same weight range $(rmin, rmax)$ which reduces both effective resistance Ω and Shortest path matrix S values of the random graph generated, hence, overall average Q-norm reduces along with its spread.
- For increase in average link resistance r, the Q-norm distributions move away from the origin indicating increase in difference between the terms of ISPP inequality. The spread of the Q-norm distributions increases with increase in r, as shown in Fig. 4.3, where standard deviation Ω increases from 0.231 for $r = 1.0$ to 1.204 for $r = 6.0$. This also

gives insight that the ISPP inequality only holds for certain set of r for a given N and p of weighted ER graphs. However, strict bounds of average link resistance r for which ISPP inequality holds may not be derived from the distribution plots and this is investigated in the Chapter 5.

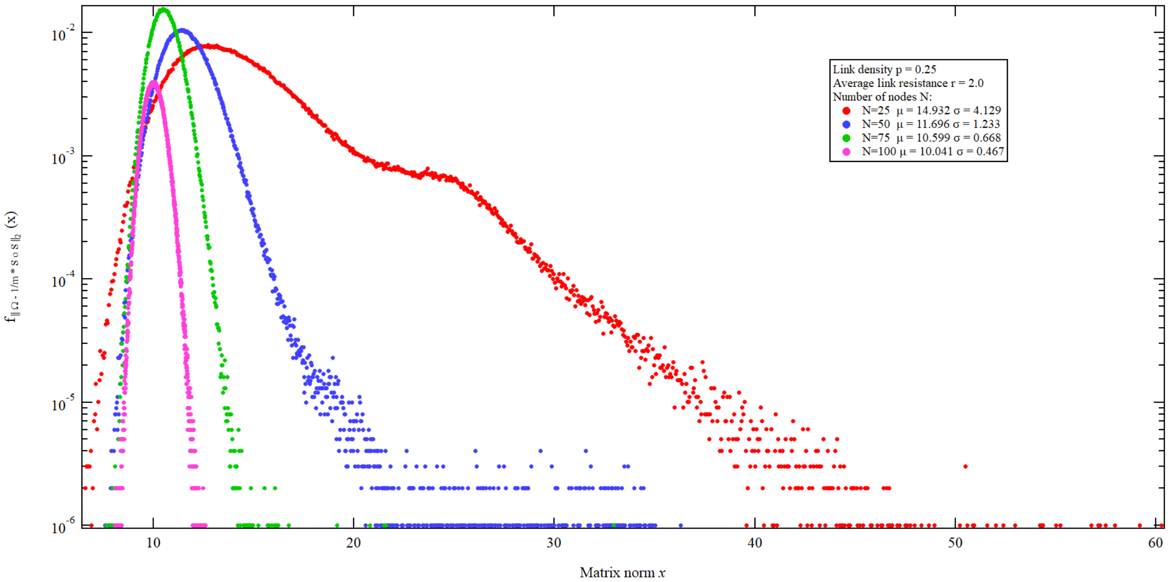


Figure 4.1: Q-norm Distribution plots of ISPP inequality for 10^6 realizations of ER graphs with $N = \{25, 50, 75, 100\}$ and link density $p = 0.25$ and average link resistance $r = 2.0$ for variation in N.

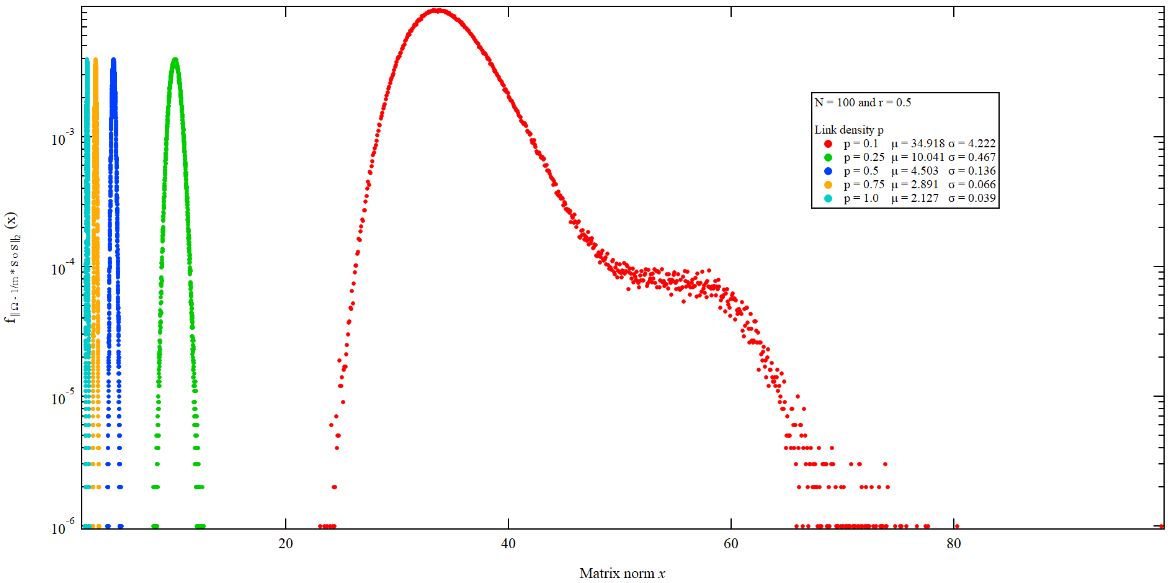


Figure 4.2: Q-norm Distribution plots of ISPP inequality for 10^6 realizations of ER graphs with $N = 100$ and link density $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$ and average link resistance $r = 2.0$ for variation in p.

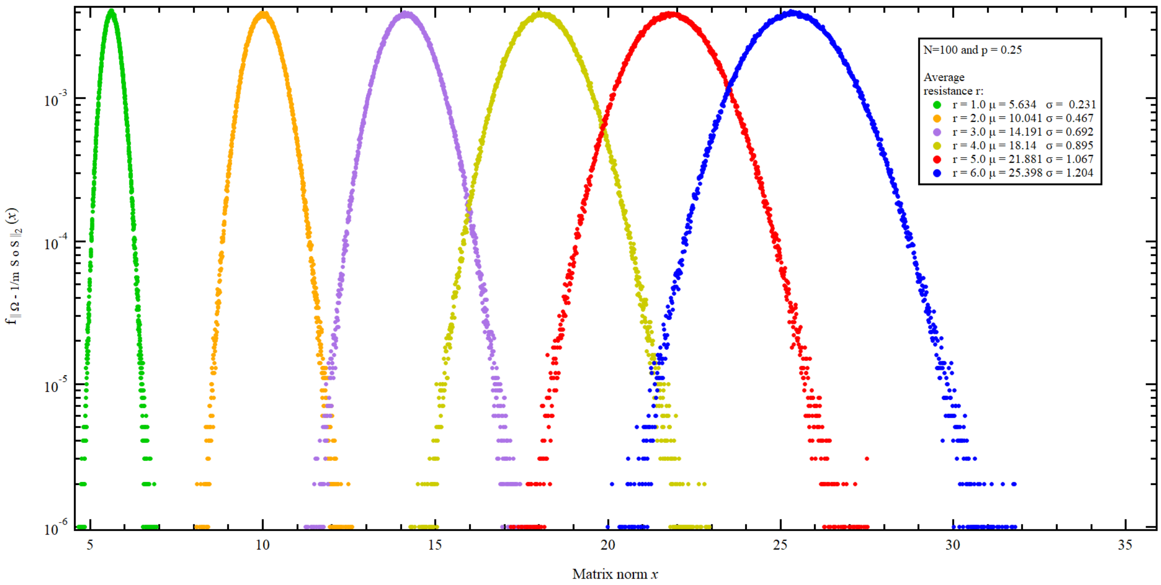


Figure 4.3: Q-norm Distribution plots of ISPP inequality for 10^6 realizations of ER graphs with $N = 100$ and link density $p = 0.25$ and average link resistance $r = \{1.0, 2.0, 3.0, 4.0, 5.0, 6.0\}$ for variation in r.

4.4. Conclusion

The Q-norm analysis on the weighted ER graphs showed that the ISPP bounds hold for weighted ER graphs with higher probability for higher N and p and the applicability of ISPP for weighted graphs is also dependent on the input parameters of ER graphs N, p and r and the Q-norm distributions suggests a possible relation between the input parameters of the graph and ISPP bounds. However, the Q-norm analysis doesn't give relation between the bounds and the input parameters of the graph. This analysis acted as an impetus to carry out the further analysis in the thesis to derive the relation showing the feasibility of implementation of ISPP for weighted ER graphs in further chapters.

5

Probability of Failure Analysis

5.1. Introduction

The Q-norm analysis revealed that ISPA is valid for weighted graphs with a certain probability and with increase in number of nodes N and link density p of weighted ER graphs the ISPA can be applicable for weighted ER graphs with increased closeness between the bounds. However, it is required to obtain the intrinsic relation between the different parameters of the weighted graphs to check the feasibility of implementation of ISPA for weighted graphs.

In this chapter, we will explore how probability of failure is used as a measure to determine the feasibility of implementation of ISPA for weighted graphs. We also explore the various insights we have gained from the analysis such as critical threshold, phase transition in probability of failure and we derive a relation between critical threshold and number of nodes N for varying link density p of the weighted ER graphs. Additionally, we will also explore the behaviour of probability of failure p_f analysis for change in minimum link resistance r_{min}.

5.2. Simulation setup

The experiment assumes the following input parameters for the weighted ER graphs generation and calculation of probability of failure.

- Number of nodes $N = \{25, 50, 75, 100\}$.
- Link density $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$.
- Average link resistance r ranging starting from 0.1 with incremental of 0.5.
- Minimum link resistance $r_{min} = 0.1$.
- Maximum link resistance $r_{max} = 2 * r - 0.1$.
- Number of realizations for each N, p and r of ER graphs $= 10^6$.

For each ER graph realization of a given combination of N, p and r, the inequality $\Omega = \frac{1}{m} * S \ o \ S$ is checked to see if it holds, else the realization is considered as failure. Similarly, out of 10^6 ER graph realizations for a given N, p and r, the number of realizations failed are collected.

Therefore,
Probability of failure p_f for a combination of N, p and $r = \frac{\text{Number of realizations failed}}{\text{Total } 10^6 \text{ realizations}}$.

5.3. Analysis

For a given link density p and number of nodes N, the probability of failure p_f is plotted for increasing average link resistance r as shown in Fig. 5.1. The initial analysis shows that for each N and p, probability of failure p_f curve follows a sigmoidal function with $(0, 1)$ transition for increase in r.

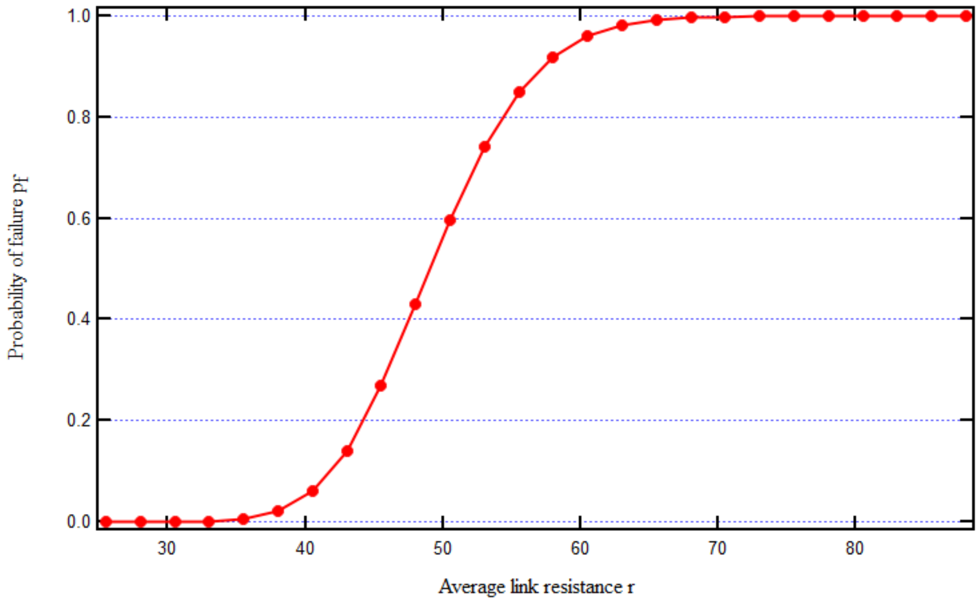


Figure 5.1: The probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = 100$ and link density $p = 1.0$.

To understand the characteristics of probability of failure p_f for increasing r and various N, it is necessary to obtain the fit closest to the sigmoidal function. In the following Subsection, the various fits closest to the phase transition of p_f vs r are considered to derive meaningful insights from them.

5.3.1. Sigmoidal Function Fitting

The various sigmoidal functions that can be the possible fits for probability of failure curve which show a transition from zero to one are presented in the following sections.

5.3.2. Tanh Hyperbolic

The tanh hyperbolic, also called as Fermi-Dirac distribution is given by

$$f_{FD}(r) = 0.5 * (1 + tanh(c * (r - r_0))) = \frac{1}{(1 + e^{-2c(r-r_0)})} \tag{5.1}$$

where r_0 is the critical threshold of the p_f curve and c is the slope factor of the p_f curve.

The Fermi Dirac distribution is a maximum entropy function which is used to study the energy states in Quantum Statistics [37]. Fermi Dirac distribution is also utilised to understand the dynamics of random processes such as network diffusion and epidemics [38]. The Fermi Dirac function fit for probability of failure p_f versus average link resistance r is shown for link density $p = 1.0$ and number of nodes $N = \{25, 50, 75, 100\}$ for weighted ER graphs in Fig.

5.2. Though the fit seems to overlap perfectly, the derivative of Fermi-Dirac distribution is required to produce a Gaussian-like distribution [39] which has not been observed in the case of probability of failure p_f curve as shown in Fig. 5.3.

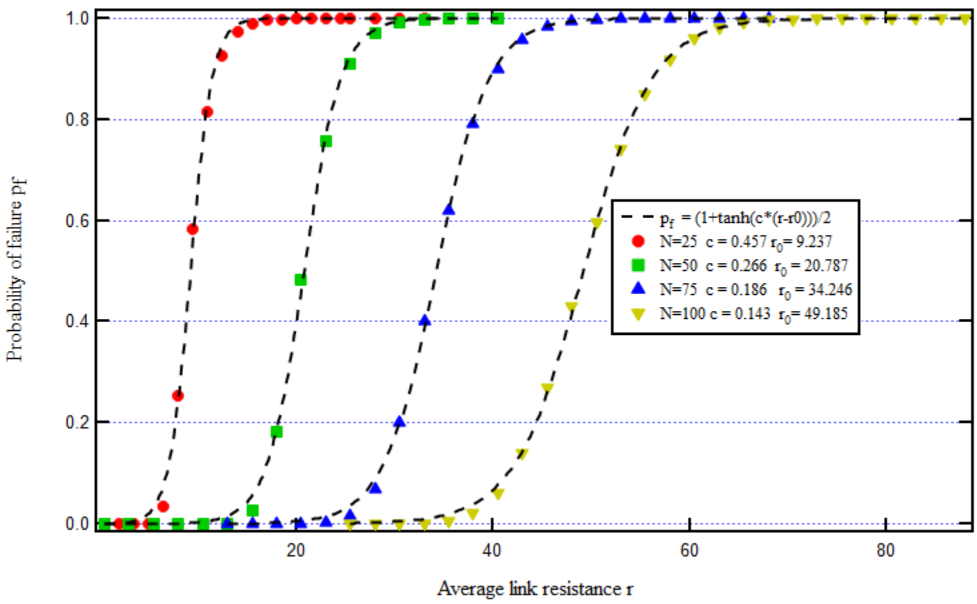


Figure 5.2: The Fermi-Dirac distribution fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 1.0$.

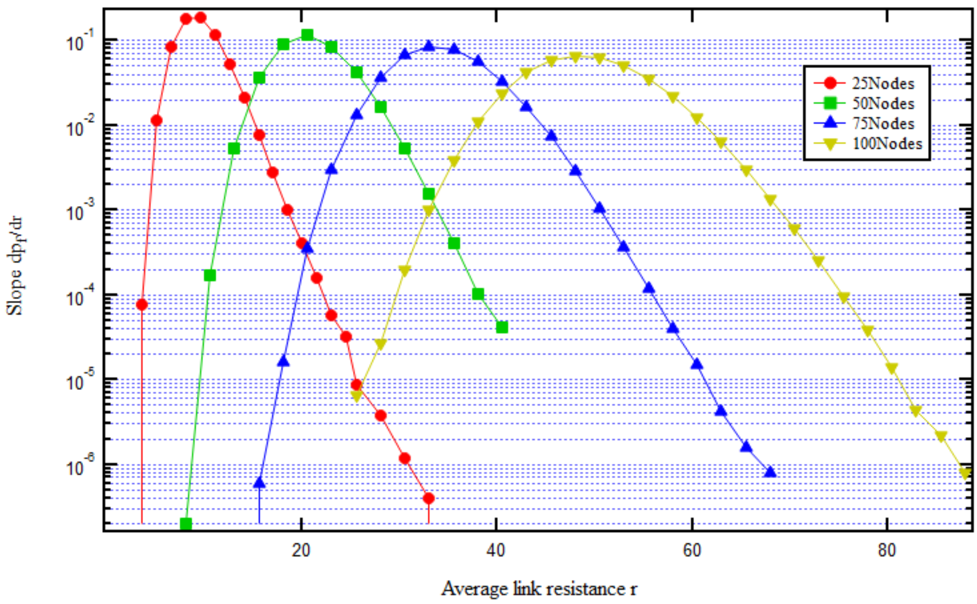


Figure 5.3: The derivative of probability of failure p_f with respect to average link resistance r which does not produce Gaussian like curve for number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 1.0$ of ER graphs.

5.3.3. Gumbel

The Gumbel distribution is an another fit that produces the zero-one transition and, theoretically, is an extremal distribution for the maximum of a set of independent and identically distributed (i.i.d) random variables [12]. The distribution that is used in the fitting is given as

$$p_f = F(r) = e^{(-e^{(-c*(r-r_0))})} \quad (5.2)$$

where r_0 is the critical threshold obtained by Gumbel fit and c is the slope factor by which probability of failure p_f increases by incrementing r by 0.5. As the derivative of a Gumbel is asymmetric in nature and probability of failure p_f curve also represents maximum of set of i.i.d. random variables, the Gumbel fit is the best fit for the p_f versus r curve. The Gumbel distribution fit for probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and various link densities $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$ is shown in Fig. 5.4 - Fig. 5.8.

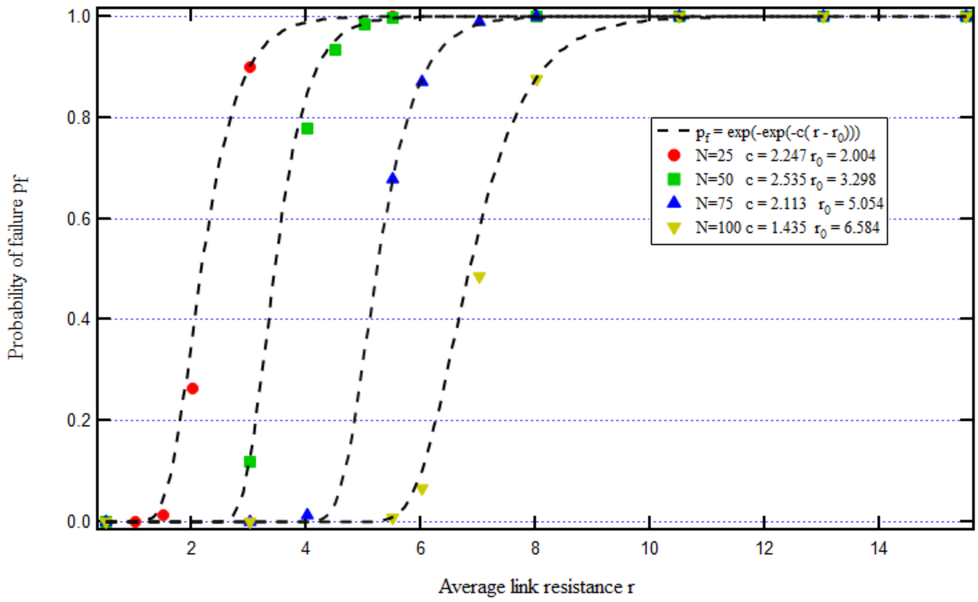


Figure 5.4: The Gumbel fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 0.1$.

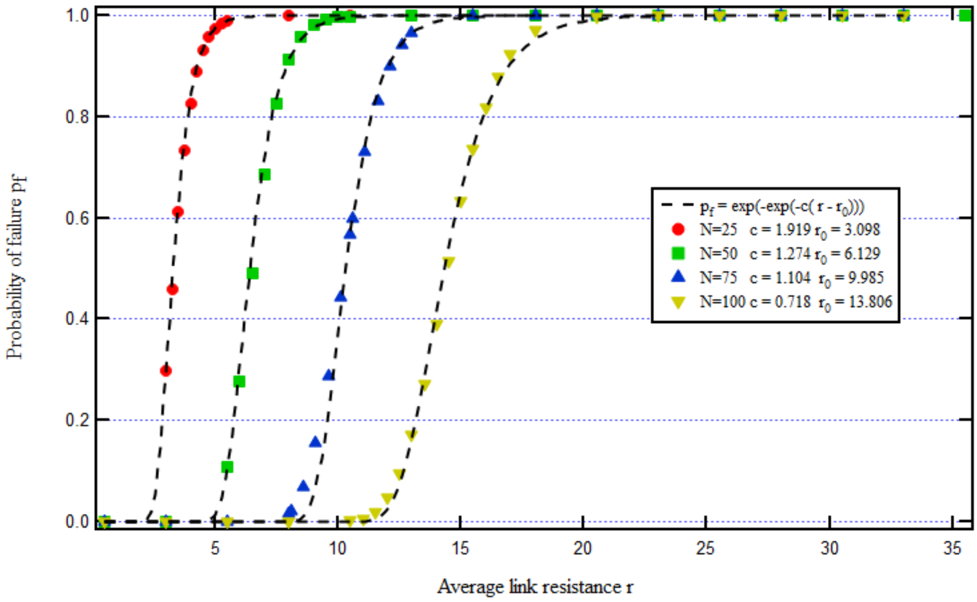


Figure 5.5: The Gumbel fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 0.25$.

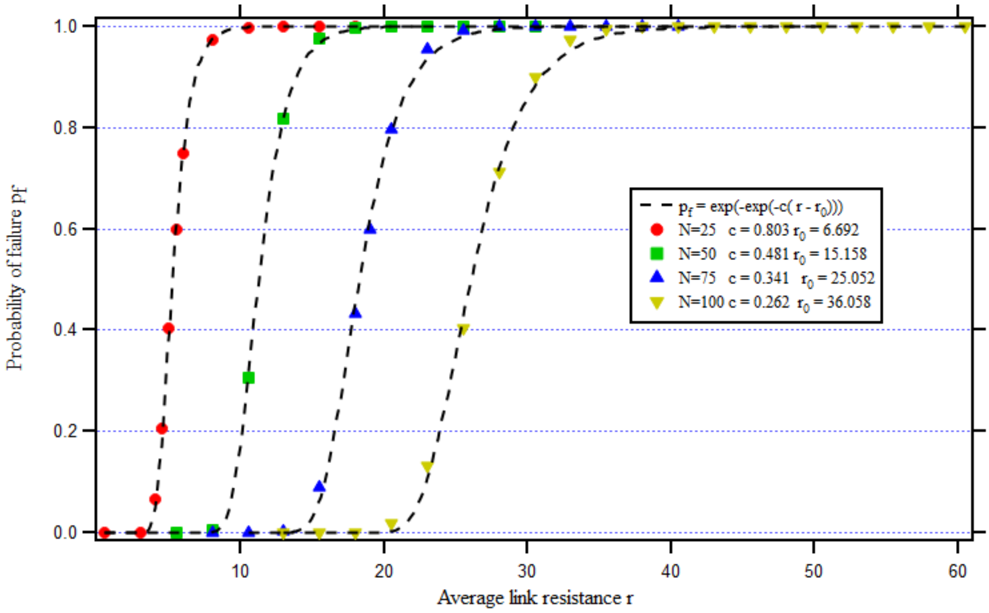


Figure 5.6: The Gumbel fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 0.5$.

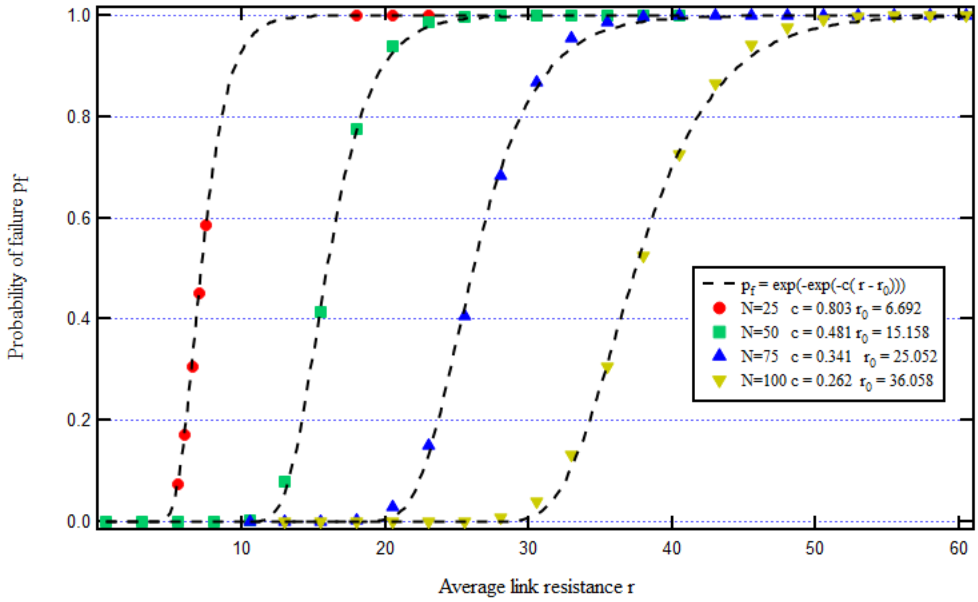


Figure 5.7: The Gumbel fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 0.75$.

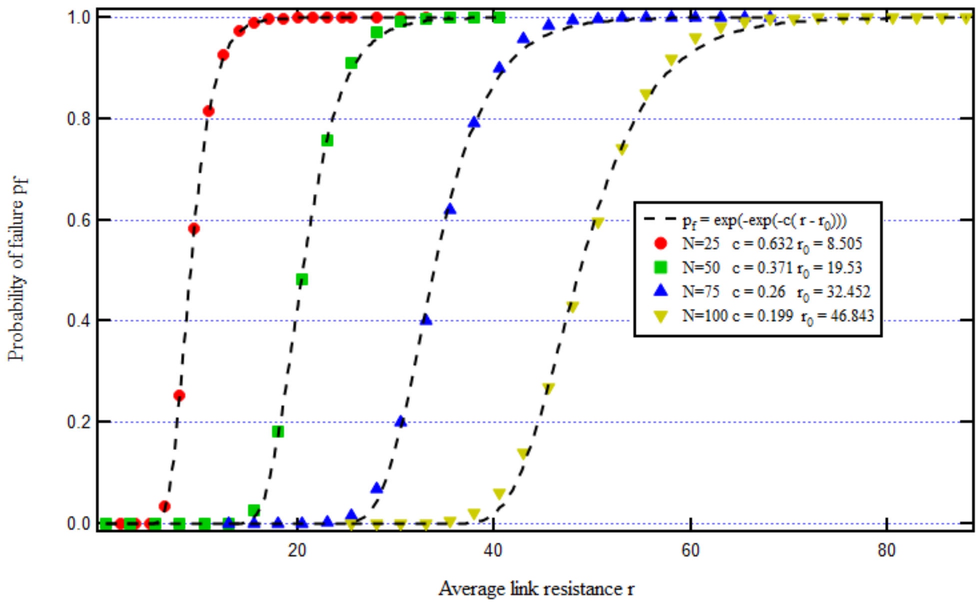


Figure 5.8: The Gumbel fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 1.0$.

5.3.4. Error function

The error function is an another function which has zero-one transition sigmoidal curve given by

$$p_f = F(r) = 0.5 \; * \; \left[1 + erf\left(\frac{r - r_0}{b}\right)\right] \tag{5.3}$$

However, the derivative of error function is a Gaussian curve [40] which clearly doesn't correlate with the asymmetry in derivative of probability of failure p_f with respect to average link resistance r curve as shown in the Fig. 5.3. Therefore, the error function is not considered for further analysis. The error function fit for p_f versus r curve for $N = \{25, 50, 75, 100\}$ and $p = 1.0$ of weighted ER graphs is shown in Fig. 5.9.

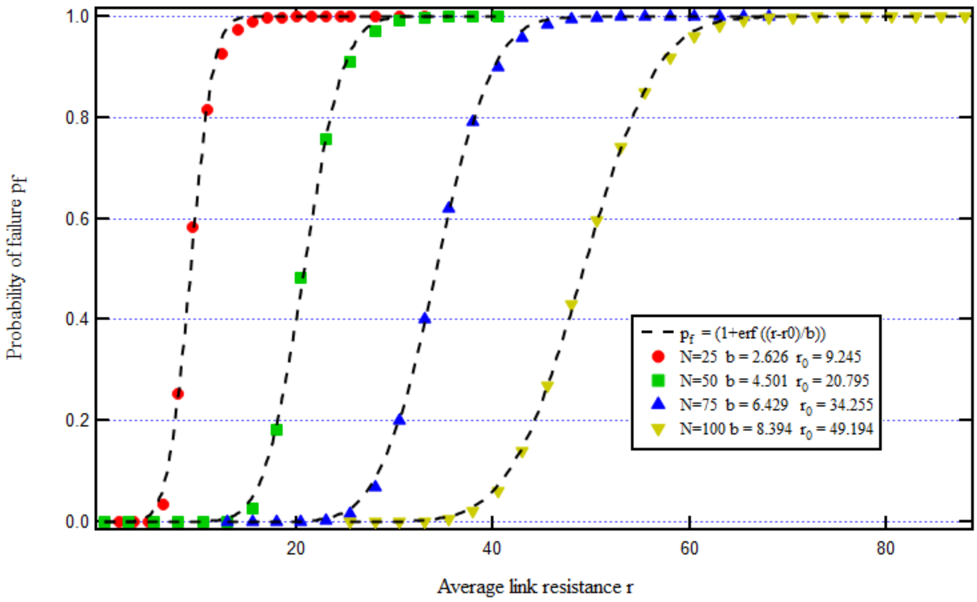


Figure 5.9: The error function fit of probability of failure p_f versus average link resistance r for weighted ER graphs with number of nodes $N = \{25, 50, 75, 100\}$ and link density $p = 1.0$.

Observations

The following are the inferences derived from the different sigmoidal fits for the probability of failure p_f versus average link resistance r plots for various number of nodes N and link density p of the ER graphs.

- Gumbel fit is the best fit among the three analysed fits; it is an extremal distribution for the maximum of set of i.i.ds for p_f versus r curve. Also, the derivative of p_f with respect to r also points out asymmetry ruling out Fermi-Dirac and error function fits from consideration even though they fit p_f vs r nicely.
- The critical threshold gives the measure of feasibility of ISPP implementation for weighted graphs which can be computed in terms of input configuration parameters of ER graphs such as N and p. The critical threshold r_0 obtained by Gumbel fit is considered for further computation explained in Subsection 5.3.5
- For a given link density p of ER graphs, the Gumbel fitting constants r_0 and c increase with increase in N corresponding to the shift of pf versus r curve away from origin towards the positive x-axis which in turn shows that the range of r for which ISPP holds increases with increase in N.
- The phase transition of probability of failure p_f is observed to be sharply increasing with increase in N. The slope factor c can provide further insights on how fast the phase transition occurs which will be discussed in Section 5.3.6.

5.3.5. Critical threshold r_0

The phase transition is the phenomenon by which a system changes its behaviour at a critical moment. This phenomenon is found in different systems of nature such as the liquid-gas phase transition and the magnetic field displacement by pressure [37]. Similar behaviour is discovered in random graphs by Erdos Renyi in 'On the evolution of random graphs' [41] where the connectivity of the graph changes from disconnectedness to connected graph by varying link density p of the graph. The critical link density $p_c = \frac{logN}{N}$ is the critical threshold in random graph, hence when $p < p_c$ the random graph is almost certainly disconnected whereas for $p > p_c$ the graph is almost certainly connected. Similar phase transition is observed in failure probability p_f of ISPP implementation of weighted random graphs.

To understand this phase transition and critical threshold for ISPP bounds, probability of failure p_f is plotted against r for different N and presented in Fig. 5.4 - Fig. 5.8. The critical threshold is the average link resistance r for which the failure probability is approximately 50%. It can be observed that, with increase in N, the critical threshold r_0 increases for a given link density p indicating a possibility of relation between r_0 and N for varying link densities. Also, when $r < r_0$, ISPP holds in most of the cases and when r is very much greater than r_0 ISPP fails for any N and p of the weighted ER graphs. Hence r_0 can be utilised to check the feasibility of implementation of ISPP for weighted graphs by presenting it in terms of input parameters of weighted ER graphs.

The critical threshold r_0 obtained by Gumbel fitting is plotted against N for varying p to find a relation between r_0, N and p, presented in Fig. 5.10. The r_0 and N follow a linear relation in log-log scale and fitting constants a and b are also presented in Fig. 5.10. The relation between r_0 and N can be given as

$$log(r_0) = a(p) \; + \; b(p) \; * \; log(N) \text{ for } p = \{0.1, 0.25, 0.5, 0.75, 1.0\} \tag{5.4}$$

Modifying the equation,

$$r_0 = 10^{a(p)} * N^{b(p)} \tag{5.5}$$

where $a(p)$ and $b(p)$ are the fitting constants which vary with link density p and are presented in Fig 5.10.

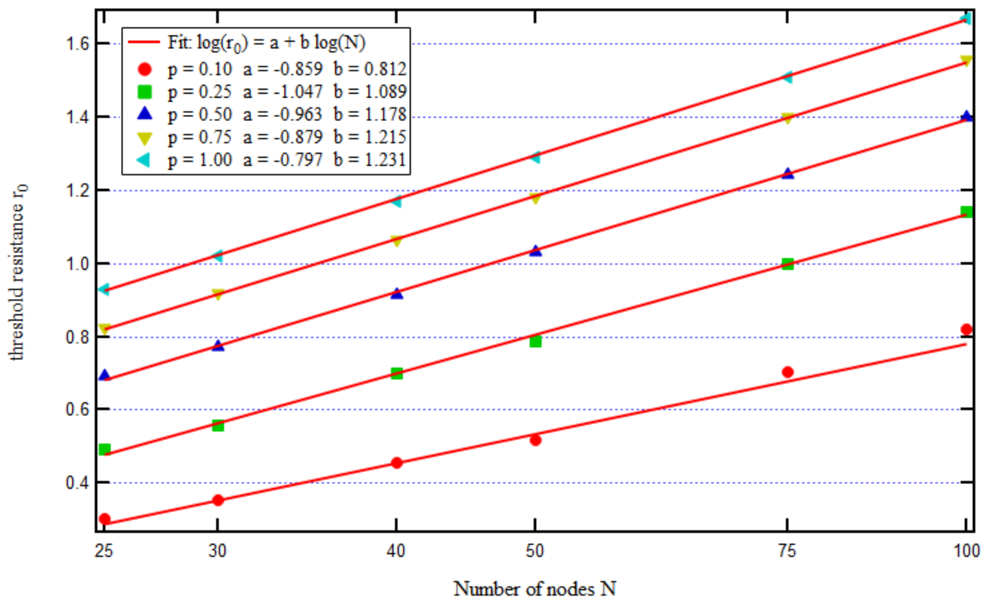


Figure 5.10: The critical threshold r_0 obtained by Gumbel fit is plotted against number of nodes N for various link densities $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$.

5.3.6. Slope factor c

The slope factor c of the Gumbel fit gives the rate at which the probability of failure p_f increases with a constant increase in r, for increase in r of 0.5 which has been plotted in Fig. 5.11. The slope factor c exhibits a linear relation with number of nodes N for various link densities of weighted ER graphs given by

$$log(c) = a(p) + b(p) * log(N) \tag{5.6}$$

Rewriting it,

$$c = 10^{a(p)} * N^{b(p)} \tag{5.7}$$

where $a(p)$ and $b(p)$ are fitting constants which vary with link density p and the values of these constants are provided in Fig. 5.11.

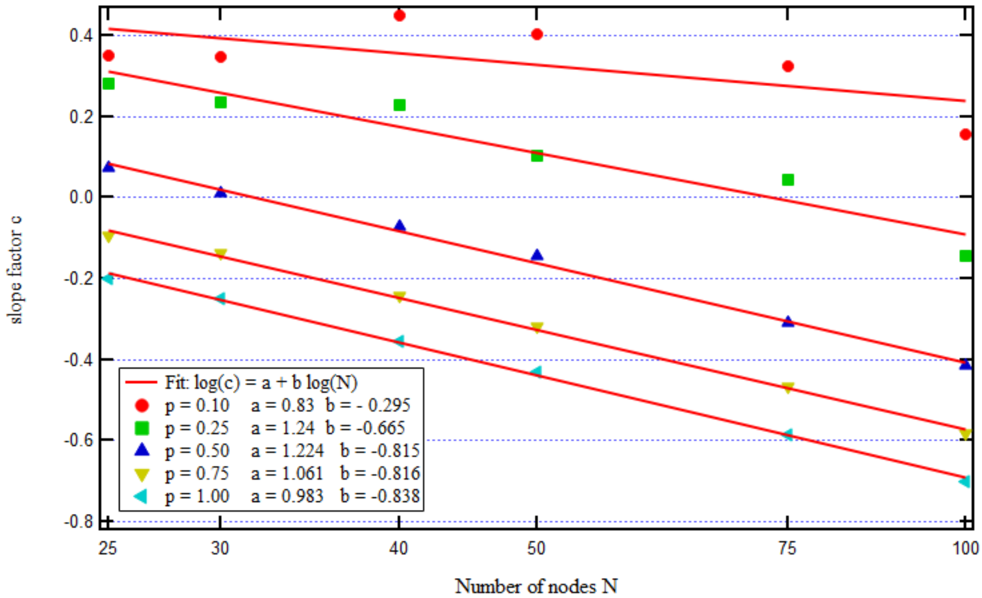


Figure 5.11: The slope factor c obtained by Gumbel fit is plotted against number of nodes N for various link densities $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$.

5.3.7. Effect of minimum link resistance r_{min} on the phase transition

The critical threshold r_0 and slope factor c relations computed so far take the assumption of minimum link resistance $r_{min} = 0.1$. But the shortest paths computation depends on both topology as well as link weight structure as mentioned in [42]. Therefore, by varying r_{min}, we observe the changes in phase transition curve between p_f and r. The aim of this study is to show that varying minimum link resistance r_{min} affects the phase transition curve.

The simulations so far run are extended for $r_{min} = \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ for $N = 25, p = 0.25$ and for 10^6 realizations for each N, p, r and r_{min} and the corresponding phase transition curve for each r_{min} is presented for p_f on y-axis and r on x-axis as shown in Fig. 5.12. It can be noted that for increase in r_{min}, the phase transition is sharply rising from 0 to 1 for $r_{min} \geq 1$. Hence when $r_{min} \geq 1$, could be the region where topology is dominant as change in link weights doesn't affect the phase transition curve of p_f. Whereas for $r_{min} < 1$, the slope of the phase transition increases with increase in r_{min} which indicates the region where link weight structure is clearly dominant. However, by varying N and p, the regions of link weight filtering and topology are dominant changes. This indicates the relations r_0 and c computed in terms of N and p can also be extended to include the r_{min} effect. However, this study requires extensive simulations and could be considered for further study to derive a relation between r_{min}, N, p and r_0. In this project, we only emphasize the role of r_{min} in phase transition curve of p_f versus r.

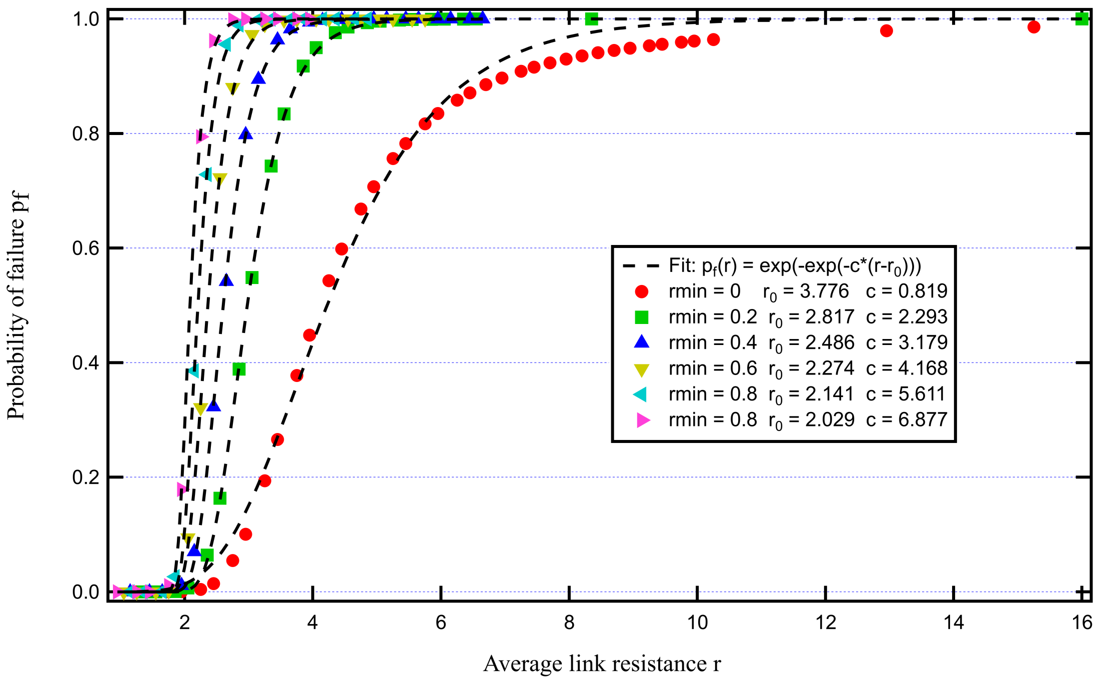


Figure 5.12: The effect of r_{min} on the phase transition curve between p_f and r for $N = 25, p = 0.25$ for weighted ER graphs for varying r and fitted by corresponding Gumbel distribution with fitting constants c and r_0.

5.4. Conclusion

The probability of failure p_f exhibits a phase transition with increasing r for varying N and p of weighted ER graphs. The probability failure p_f versus average link resistance r is closely fitted by Gumbel function and through which the critical threshold r_0 and slope factor c are expressed in terms of ER weighted graph input parameters N and p. The equation with r_0, N and p can be utilised as a measure to check the feasibility of implementation of ISPP for weighted graphs. Finally, we have shown that r_{min} chosen has an effect on phase transition curve between p_f and r.

6

Hopcount distribution analysis

6.1. Introduction

We have seen that ISPP bounds hold for weighted graphs for a certain range of link weights and a feasibility condition has been derived based on the critical threshold r_0 and input parameters N, p and r of weighted ER graphs. In this chapter, we aim at understanding the characteristics of solutions possible for ISPA through hopcount distribution analysis.

Hopcount between any two nodes of a network gives the number of traversed nodes in the shortest path between the node pair (see Section 2.5.2). The hopcount is considered as an important measure in the contemporary IP networks, especially, for QoS measures such as packet delay, jitter and packet loss which are dependent on the number of traversed routers in the shortest path – hopcount, rather than the length of the shortest path [43]. Hence, the objective of the analysis is to understand the characteristics of hopcount distribution of shortest paths of the solutions obtained through ISPA.

6.2. Simulation setup

Initially, weighted ER graphs are generated for each combination of (N, p, r) for 10^6 realizations for the following set of values of input parameters.

- Number of nodes $N = \{25, 50, 75, 100\}$.
- Link density $p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$.
- Average link resistance r ranging starting from 0.1 with incremental of 0.5.
- Minimum link resistance $r_{min} = 0.1$.
- Maximum link resistance $r_{max} = 2 * r - 0.1$.

For each realization of ER graph, difference $\Omega - \frac{1}{m}SoS$ is computed for ISPA inequality (see detailed procedure in Section 4.2). If the difference is greater than or equal to zero, hopcount of the generated ER graph realization is computed and stored in an array. This procedure is repeated for 10^6 realizations of weighted ER graphs for a given set of input parameters (N, p, r) and the hopcount distribution obtained for all the realizations are collected in the same array. The hopcount distribution data consists of two arrays. One being the possible hopcount values possible for a given number of Nodes N. For example, for $N = 25$, the possible hopcount values are $[1, 2, 3, \ldots 24]$. The second array is the frequencies corresponding the hopcount values which are obtained through the extensive simulations. The code developed for hopcount distribution analysis can be found in Appendix B.

6.3. Analysis

By plotting the frequency with respect to the hopcount values, we obtain Fig. 6.1 for weighted graph simulations with $N = 25, p = 0.25$ and $r = 1.0$ which suggests that the hopcount distribution approximates Poisson distribution provided the frequencies are normalised to probability and mean of the distribution that are computed as (6.1) and (6.2) respectively, due to the fact that the hopcount distribution in Fig. 6.1 has events which are discrete and independent.

$$\text{Normalised frequency of a hopcount value} = \frac{\text{Frequency corresponding the hopcount value}}{\text{Sum of the frequencies for all the hopcount values}} \tag{6.1}$$

$$\text{Mean of the hopcount distribution} = \sum \text{Hopcount value} \times \text{Its corresponding normalised frequency} \tag{6.2}$$

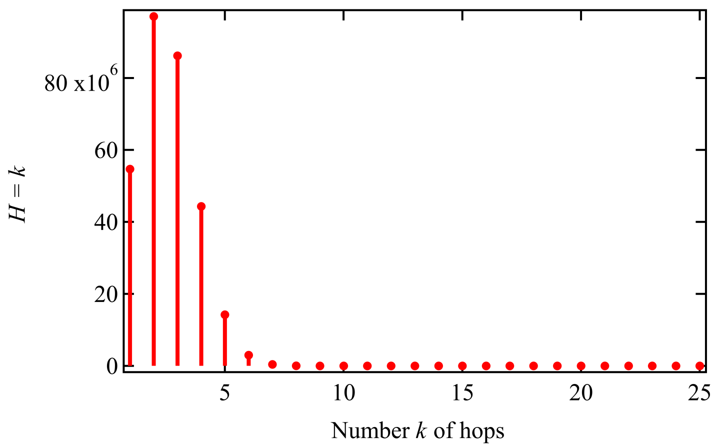


Figure 6.1: Hopcount distribution for weighted ER graph with $N = 25, p = 0.25$ and $r = 1.0$

6.3.1. Poisson Distribution

A discrete random variable X is said to have Poisson distribution if its distribution is given by [44]

$$Pr(X = k) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{6.3}$$

where λ = Mean or Expected value of X and k = Number of occurrences of event.

Poisson distribution is utilised in various discrete real-life computations. One such example is the call blocking probability in mobile communications network which is computed to model the capacity of the network provided a mean of λ calls arrive per hour and all the calls are independent [45].

By calculating the Poisson distribution values for each hopcount with mean $\lambda = 2.59$ using (6.2) for each of the hopcount values, we obtain the Poisson distribution values as shown in Table 6.1. By plotting the normalised hopcount distribution and its corresponding Poisson distribution as shown in Fig. 6.2, we show that ISPA provides solutions which meets the constraints provided through the demand matrices D for the weighted random graphs such that r is considerably less than r_0 and the ISPA inequality holds.

Hopcount	Frequency	Normalised frequency	Poisson distribution values
1	54704418	0.182	0.194
2	97102223	0.324	0.252
3	86184426	0.287	0.217
4	44314315	0.148	0.141
5	14235294	0.047	0.073
6	2990163	0.010	0.031
7	424325	0.001	0.012
8	41741	0.000	0.004
9	2940	0.000	0.001
10	149	0.000	0.000
11	5	0.000	0.000
12	1	0.000	0.000
13	0	0.000	0.000
14	0	0.000	0.000
15	0	0.000	0.000
16	0	0.000	0.000
17	0	0.000	0.000
18	0	0.000	0.000
19	0	0.000	0.000
20	0	0.000	0.000
21	0	0.000	0.000
22	0	0.000	0.000
23	0	0.000	0.000
24	0	0.000	0.000
25	0	0.000	0.000

Table 6.1: Hop count distribution of 10^6 weighted ER graphs for $N = 25, p = 0.25$ and $r = 1.0$ along with their normalised frequencies and corresponding Poisson distribution values.

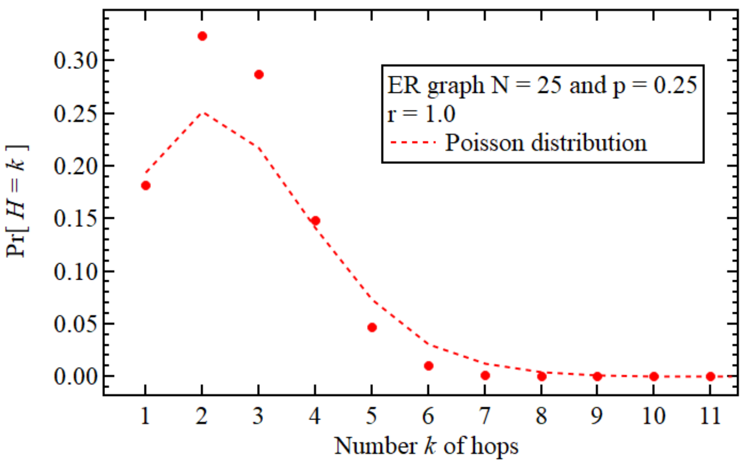


Figure 6.2: Hopcount distribution of weighted ER graphs with $N = 25, p = 0.25$ and $r = 1.0$ with mean $\lambda = 2.59$ presented in terms of normalised frequencies and their corresponding Poisson values.

6.4. Conclusion

The aim of the analysis was to study the hopcount characteristics of the solutions obtained through ISPA for weighted graphs, as hopcount is termed as an important parameter for service differentiation in contemporary IP networks. We conclude that the hopcount distribution of solutions obtained through ISPA can be approximated by Poisson distribution with mean λ and ISPA provides solutions which meets the constraints provided r is considerably less than r_0 and the inequality holds.

7

Validation of ISPA

7.1. Introduction

We have, so far, studied the various properties of ISPP bounds for weighted graphs such as Q-norm analysis, probability of failure analysis and hopcount distribution analysis through which we explored the nature of ISPP bounds for weighted ER graphs and derived a relation for feasibility of implementation of ISPA. In this chapter, we aim at validating the ISPA to obtain non-negative weighted Adjacency from a given demand matrix D and to verify the feasibility condition derived between p_f and r.

7.2. Generation of demand Matrices

The main challenge in validating ISPA is the conversion of non-simplex distance matrix to simplex in step 4 of ISPA (see Section 3.7) which is yet unknown. Hence, it is not yet possible to derive simplex distance matrix for any random demand matrix D. So we compute the demand matrix D from the effective resistance matrix Ω which is a squared Euclidean distance matrix and by taking square root of Ω we obtain distance matrices which are simplex in nature [46]. The procedure is as follows:

- Generate effective resistance matrix Ω for any N, p and r of a weighted ER graph as shown in Section 2.5.3.
- To verify step 3 of ISPA, add a set of small random elements e within a range of $(emin, emax)$ to the lower triangle of the Ω in order to obtain modified effective resistance matrix given by Ω'. Adding e to all the elements of Ω may result in a non-simplex distance matrix while solving for ISPA, hence e is added only to the lower triangle of Ω such that ISPA can resolve the matrix to simplex distance matrix.
- The demand matrix D is obtained as $D = \frac{N}{\sqrt{2}} * \sqrt{\Omega'}$.

Through this procedure, we ensure that the demand matrix always returns a non-negative weighted Adjacency through ISPA. The code for generating demand matrices is provided in the Appendix C.

7.3. Verification of feasibility relation

The feasibility condition derived between p_f and r in Subsection 5.3.5 for the ISPA lower bound also needs to be verified for which the following procedure has been followed:

- For a given demand matrix D, once a non-negative weighted Adjacency $\tilde{A}$ is obtained, the terms N, p and r are computed from $\tilde{A}$.
- For solution obtained through ISPA, critical threshold r_0 is computed from the feasibility condition and Q-norm is computed to check if the ISPA inequality holds. This procedure is repeated for 1000 instances of demand matrices generated for the input parameters of $N = 25, p = 0.25$ and increasing r.
- The p_f obtained by ISPA is compared with the feasibility condition derived between p_f and r to verify the precision of the conditions derived.

The code for verifying the feasibility condition can be found in the Appendix C. The procedure followed to verify the feasibility condition through ISPA is given in Fig. 7.1. Table 7.1 gives the comparison of the feasibility condition derived to that of actual implementation. It follows from Table 7.1 that p_f computed from probability of failure analysis correlates with the actual p_f computed from the ISPA solution for randomly generated demand matrices for a given N, p and r of the weighted ER graphs.

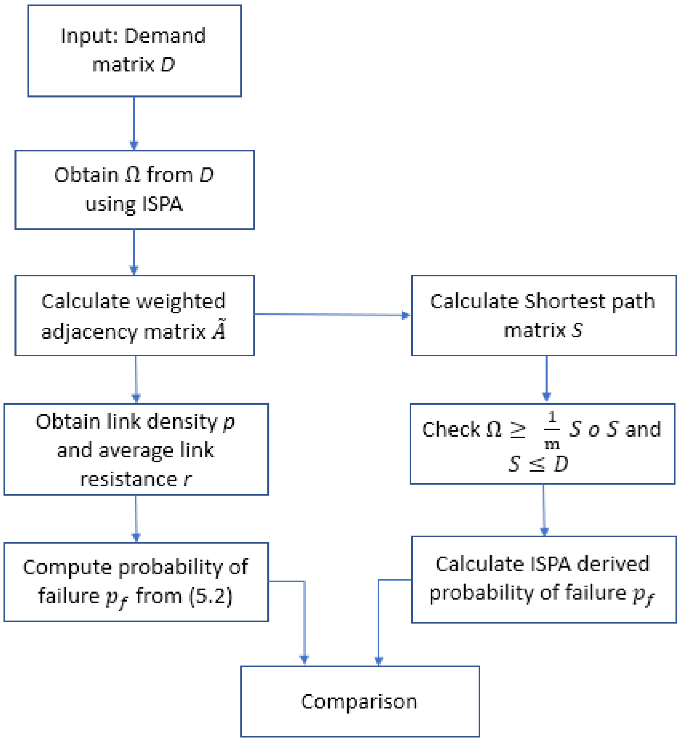


Figure 7.1: Procedure followed to validate feasibility condition derived between p_f and r in (5.2) through ISPA for weighted ER graphs.

r	r_0	p_f calculated from (5.2)	p_f calculated from ISPA solution
1.05	3.098	0	0
2.05	3.098	0.005	0.004
3.05	3.098	0.331	0.332
4.05	3.098	0.851	0.849
5.05	3.098	0.976	0.978
6.05	3.098	0.996	0.996
7.05	3.098	1	1

Table 7.1: The comparison of p_f computed from (5.2) to that of p_f computed from ISPA for 1000 demand matrices of weighted ER graphs with $N = 25, p = 0.25, r_{min} = 0.1$ and increasing r.

7.4. Conclusion

The aim of this Section is to check the feasibility of ISPA implementation by generating demand matrices which can be resolved into simplex distance matrices and verify the feasibility conditions derived in Chapter 5. We have successfully generated demand matrices which gives non-negative weighted adjacency matrices as solutions and verified the feasibility condition for weighted ER graphs for given input parameters.

8

Conclusions and future work

In this chapter, we summarize our work and propose some suggestions for future work. We start with the main conclusions of the thesis in Section 8.1 followed by recommendations for future work in Section 8.2.

8.1. Conclusions

The aim of the thesis was

1. To understand the nature of ISPP bounds for weighted random graphs with uniformly generated link weights through various analyses such as Q-Norm distributions, probability of failure and hop count distributions.
2. To check the feasibility of ISPA implementation for weighted random graphs based on the relation between various parameters of the graph obtained by the analysis of the nature of ISPP bounds.
3. To implement ISPA for any randomly generated demand matrices so as to obtain a non-negative solution and verify the feasibility for ISPA solutions.

We started by explaining the various metrics of graph theory and random graph models, reviewing various works of peers to solve network resource allocation problem and understanding the methodology of ISPA and establishing the need of the thesis. We have performed the Q-norm analysis in Chapter 4, where we concluded that Q-norm distributions mostly followed binomial distributions and understood the variation of Q-norm distributions for variation in input parameters of the ER graph such as N, p and r.

In Chapter 5, we computed p_f versus r sigmoidal transition graphs for ISPA inequality and concluded Gumbel is the most appropriate fit for the sigmoidal curves, from which critical threshold r_0 and slope factor c are expressed in terms of input parameters N, p and r of the ER graph. We continued our research to understand that the hopcount distribution of solutions obtained for ISPA inequality can be approximated by Poisson distribution with mean λ in Chapter 6.

Finally, in Chapter 7, we have generated demand matrices that can be resolved into simplex distance matrices in the algorithm and we were able to verify the feasibility condition derived in Chapter 5 with ISPA solutions obtained from the demand matrices.

8.2. Future work

Based on the results and conclusions, the following future work is suggested.

- We have derived a relation between r_0, c with a certain set of input parameters of ER graphs such as $N = \{25, 50, 75, 100\}, p = \{0.1, 0.25, 0.5, 0.75, 1.0\}$. The work can be extended to include a wide range of input parameters such that accurate relations can be obtained for N and p of the ER graphs through extensive simulations.
- We have also made an important assumption that minimum link resistance $r_{min} = 0.1$ of the link weights to derive the relation between p_f and r. In Chapter 5, it has been shown that the phase transition of p_f is also dependent on r_{min}. It is suggestive to include the effect of r_{min} in the relations (5.4) and (5.6) for completeness which can also be achieved through extensive simulations.
- The demand matrix generation in Chapter 7 is performed through Ω, as conversion of non-simplex distance matrix to simplex in steps of ISPA algorithm is unknown. A heuristic approach can be developed to convert any non-simplex distance matrix to simplex for improving the applicability of the algorithm for wide range of domains and improve the utility of the algorithm.

References

[1] Ning Wang et al. “An overview of routing optimization for internet traffic engineering”. In: *IEEE Communications Surveys Tutorials* 10.1 (2008), pp. 36–56. DOI: 10.1109/COMST.2008.4483669.

[2] Athanassios A Androutsos and Theodore K Apostolopoulos. “A demand-based approach to optimal resource allocation for network services with Quality of Service (QoS) requirements”. In: *2006 IEEE First International Workshop on Bandwidth on Demand*. IEEE. 2006, pp. 49–56.

[3] Jocelyne Elias. “Optimal Resource Allocation in Networks: Optimization, Game Theoretical Models, and Algorithms”. In: (2018).

[4] Bernard Fortz and Mikkel Thorup. “Internet traffic engineering by optimizing OSPF weights”. In: *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*. Vol. 2. IEEE. 2000, pp. 519–528.

[5] Michal Pióro and Deep Medhi. *Routing, flow, and capacity design in communication and computer networks*. Elsevier, 2004.

[6] Andreas Bley. “Finding small administrative lengths for shortest path routing”. In: *Proc. of 2nd International Network Optimization Conference, Lisbon, Portugal*. Citeseer. 2005, pp. 121–128.

[7] M Ericsson, Mauricio G. C. Resende, and Panos M. Pardalos. “A genetic algorithm for the weight setting problem in OSPF routing”. In: *Journal of combinatorial optimization* 6.3 (2002), pp. 299–333.

[8] P Van Mieghem, K Devriendt, and R Noldus. “Inverse shortest path problem: Flow-based resource management for path networks”. unpublished. 2021.

[9] Piet Van Mieghem. *Graph Spectra for Complex Networks*. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511921681.

[10] Paul Erdös and Alfréd Rényi. “On random graphs Publ”. In: *Math. debrecen* 6 (1959), pp. 290–297.

[11] Edgar N Gilbert. “Random graphs”. In: *The Annals of Mathematical Statistics* 30.4 (1959), pp. 1141–1144.

[12] Piet Van Mieghem. *Performance analysis of communications networks and systems*. Cambridge University Press, 2009.

[13] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. USA: Prentice-Hall, Inc., 1993. ISBN: 013617549X.

[14] Ayşegül Altın et al. “Intra-domain traffic engineering with shortest path routing protocols”. In: *Annals of Operations Research* 204.1 (2013), pp. 65–95.

[15] M Pioro et al. “On OSPF related network optimisation problems”. In: *P r oceed i n gs o fIFIPA TM IP, Ilkley UK* (2000).

[16] Bernard Fortz and Mikkel Thorup. “Optimizing OSPF/IS-IS weights in a changing world”. In: *IEEE journal on selected areas in communications* 20.4 (2002), pp. 756–767.

[17] Bernard Fortz and Mikkel Thorup. “Increasing internet capacity using local search”. In: *Computational Optimization and Applications* 29.1 (2004), pp. 13–48.

[18] A Jüttner et al. “On solvability of an OSPF routing problem”. In: *Ericsson Traffic Laboratory, Budapest, Hungarya, Dept. of Communication Systems, Lund Institute of Technology, Sweden* (2000).

[19] M. Kodialam and T. V. Lakshman. “Minimum interference routing with applications to MPLS traffic engineering”. In: *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*. Vol. 2. 2000, 884–893 vol.2. DOI: 10.1109/INFCOM.2000.832263.

[20] Natalie Degrande et al. “Inter-area traffic engineering in a differentiated services network”. In: *Journal of Network and Systems Management* 11.4 (2003), pp. 427–445.

[21] Simon Balon, Fabian Skivée, and Guy Leduc. “How well do traffic engineering objective functions meet TE requirements?” In: *International Conference on Research in Networking*. Springer. 2006, pp. 75–86.

[22] Luciana S Buriol et al. “A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing”. In: *Networks: An International Journal* 46.1 (2005), pp. 36–56.

[23] Yufei Wang, Zheng Wang, and Leah Zhang. “Internet traffic engineering without full mesh overlaying”. In: *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*. Vol. 1. 2001, 565–571 vol.1. DOI: 10.1109/INFCOM.2001.916782.

[24] F. Blanchy, L. Melon, and G. Leduc. “Routing in a MPLS network featuring preemption mechanisms”. In: *10th International Conference on Telecommunications, 2003. ICT 2003.* Vol. 1. 2003, 253–260 vol.1. DOI: 10.1109/ICTEL.2003.1191228.

[25] A. Elwalid et al. “MATE: MPLS adaptive traffic engineering”. In: *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*. Vol. 3. 2001, 1300–1309 vol.3. DOI: 10.1109/INFCOM.2001.916625.

[26] Kaj Holmberg and Di Yuan. “Optimization of Internet Protocol network design and routing”. In: *Networks* 43.1 (2004), pp. 39–53. DOI: https://doi.org/10.1002/net.10102. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.10102. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.10102.

[27] A. Farago, Aron Szentesi, and Balázs Szviatovszki. “Inverse optimization in high-speed networks”. In: *Discrete Applied Mathematics* 129 (June 2003), pp. 83–98. DOI: 10.1016/S0166-218X(02)00235-4.

[28] Changyong Zhang and Robert Rodosek. “Modelling and Constraint Hardness Characterisation of the Unique-Path OSPF Weight Setting Problem”. In: *Computational Science – ICCS 2005*. Ed. by Vaidy S. Sunderam et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 804–811.

[29] John E Beasley and Nicos Christofides. “An algorithm for the resource constrained shortest path problem”. In: *Networks* 19.4 (1989), pp. 379–394.

[30] C St JA Nash-Williams. “Random walk and electric currents in networks”. In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Vol. 55. 2. Cambridge University Press. 1959, pp. 181–194.

[31] Miroslav Fiedler. “Some characterizations of symmetric inverse M-matrices”. In: *Linear Algebra and its Applications* 275-276 (1998). Proceedings of the Sixth Conference of the International Linear Algebra Society, pp. 179–187. ISSN: 0024-3795. DOI: https://doi.org/10.1016/S0024-3795(97)10022-2. URL: https://www.sciencedirect.com/science/article/pii/S0024379597100222.

[32] Miroslav Fiedler. *Matrices and graphs in geometry*. 139. Cambridge University Press, 2011.

[33] Piet Van Mieghem, Karel Devriendt, and H Cetinay. “Pseudoinverse of the Laplacian and best spreader node in a network”. In: *Physical Review E* 96.3 (2017), p. 032311.

[34] Karel Devriendt. “Effective resistance is more than distance: Laplacians, Simplices and the Schur complement”. In: *arXiv preprint arXiv:2010.04521* (2020).

[35] Karel Devriendt, Samuel Martin-Gutierrez, and Renaud Lambiotte. “Variance and covariance of distributions on graphs”. In: *arXiv preprint arXiv:2008.09155* (2020).

[36] Charles Chapman Pugh and CC Pugh. *Real mathematical analysis*. Vol. 2011. Springer, 2002.

[37] Peter S Riseborough. *Statistical Mechanics*. World Scientific, 2020.

[38] Liubov Tupikina. “Dynamics on networks. Case of Heterogeneous Opinion Status Model”. In: *arXiv preprint arXiv:1708.01647* (2017).

[39] M Godfrey. “The TANH transformation”. In: *Information Systems Laboratory, Stanford University. Published by Stanford Univ., CA* (2009).

[40] Eric W. Weisstein. *Erf. From MathWorld–A Wolfram Web Resource.* URL: https://mathworld.wolfram.com/Erf.html.

[41] Paul Erdös and Alfréd Rényi. “On the evolution of random graphs”. In: *The structure and dynamics of networks*. Princeton University Press, 2011, pp. 38–82.

[42] Huijuan Wang and Piet Van Mieghem. “Shifting the Link Weights in Networks”. In: *2007 Second International Conference on Communications and Networking in China*. IEEE. 2007, pp. 67–71.

[43] Piet Van Mieghem, Gerard Hooghiemstra, and Remco van der Hofstad. “A scaling law for the hopcount in internet”. In: *Delft University of Technology, report* 2000125 (2000).

[44] Richard H McCuen. “-Probability Distributions for Discrete Random Variables”. In: *Probability, Statistics, and Reliability for Engineers and Scientists*. CRC Press, 2016, pp. 142–167.

[45] Nadra Abdalla and Richard J Boucherie. “Blocking probabilities in mobile communications networks with time-varying rates and redialing subscribers”. In: *Annals of Operations Research* 112.1 (2002), pp. 15–34.

[46] Karel Devriendt and Piet Van Mieghem. “The simplex geometry of graphs”. In: *Journal of Complex Networks* 7.4 (2019), pp. 469–490.

A
Appendix

Matlab Code for the simulations done to obtain Q-Norm distribution analysis and probability of failure analysis results.

```
% Program name: simulation.m
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-03-12
%
% Generates 10^6 realizations of weighted ER graphs for input parameters
% number of nodes n, link density p and average link resistance r.
% Calcultes effective resistance matrix, Shortest path matrix and computes
% Q-norm of ISPA lower bound and thus determines probability of failure.

clc;
clear all;

% Inputs
n = 25;          % Number of nodes
p = 0.75;        % Link density
rmin = 0.1;       % Minimum link resistance

% Array initialization to store the results
nm_ar = [];      % Stores Q-norm values
pf = 0;          % Set failure cases to 0
nodes = [];      % Stores n for each realization
prob = [];       % Stores p for each realization
r_avg = [];      % Stores r for each realization
norm_save = []; % Stores average Q-norm for each realization
fail_prob = []; % Stores number of failure cases for 10^6 realizations
links = [];      % Stores number of links for each realization
r_actual = [];  % Stores r computed from the graph for each ralization


% Input rmax values for which program needs to run
for rmax = [1.9, 2.9, 3.9]

    %Intermediate arrays and variables
    l = 0;
    r_calc = 0;
    l_arr = [];
    r_calc_arr = [];


    % Number of realizations
    parfor rlz = 1:1000000

        % Generates unweighted ER graph
```

```
        A = generate_ER(n,p)

        %weights generation in [rmax,rmin] range
        weights = rmin + (rmax - rmin).*rand(n,n);
        lower_traingle = tril(A,-1);
        weight_tril = lower_triangle .* weights;

        % Weighted Adjacency matrix generation
        weight_adj = weight_tril + transpose(weight_tril);

        % Flow based weighted adjacency matrix
        flow_weight = 1./weight_adj;
        flow_weight(~isfinite(flow_weight)) = 0;

        % Generates graph with weighted adjacency
        g1 = graph(weight_adj);

        % Uncomment to plot the graph
        %plot(g1,'EdgeLabel',g1.Edges.Weight);

        % Shortest path matrix
        dist = distances(g1);

        % Degree vector
        degree_vec = sum(flow_weight,2);

        % Laplacian
        lap = diag(degree_vec) - flow_weight;

        % Omega matrix - Calls function 'resistance'
        omega = resistance(lap,n);

        % S o S calculation
        sos = dist.* dist;

        % m calculation
        inv_link_wt = 1./weight_tril;
        inv_link_wt(~isfinite(inv_link_wt)) = 0;
        m = sum(inv_link_wt,'all');

        %ISPA inequality
        diffX = omega - ((1/m).*sos);

        % If ISPA inequality holds compute Q-norm else
        % count number of failure cases
        if(all(diffX(:)>=0))
            qnorm = norm(diffX);
            nm_ar = [nm_ar qnorm];
        else
           pf = pf + 1
        end

        % Computes number of links for a realization
        % This code is not mandotary. Only for verification.
        l = sum(A(:))/2;
        l_arr = [l_arr l];

        % Computes actual r for a realization
        % This code is not mandotary. Only for verification.
        r_calc = sum(weight_tril(:))/l;
        r_calc_arr = [r_calc_arr r_calc];

    end
    % Computes average number of links, r and average q-norm
    % for 10^6 realizations.
    l_avg = mean(l_arr);
    r_actual_avg = mean(r_calc_arr);
```

```
    norm_avg = mean(nm_ar);

    % Uncomment to store Q-norm array
    % writematrix(nm_ar', '/home/sdarsi/bulk/exceldata/N='+string(n)+'/qnorm_hist.
        txt');

    % Store data for output
    nodes = [nodes n];
    prob= [prob p];
    r_avg = [r_avg ((rmax+0.1)/2)];
    norm_save = [norm_save norm_avg];
    fail_prob = [fail_prob pf] ;
    norm_avg = 0;
    links = [links l_avg];
    r_actual = [r_actual r_actual_avg];

    % Set all the values to zero after each run.
    pf = 0;
    l_avg = 0;
    r_actual_avg = 0;
    nm_ar = [];
    neg_nm_ar = [];
end

% Copy results to excel.
mymatrix(:,1) = nodes;
mymatrix(:,2) = prob;
mymatrix(:,3) = r_avg;
mymatrix(:,4) = r_actual;
mymatrix(:,5) = variance_list;
mymatrix(:,6) = links;
mymatrix(:,7) = norm_save;
mymatrix(:,8) = fail_prob;

%Output File name
str2 = "N=" + string(n) + ", p=" + string(p)+ ", rmax="+ string(rmax);

% Output excel file path
writematrix(mymatrix,'/home/sdarsi/bulk/exceldata/er_graph/'+ str2+ '.xlsx');
```

Matlab Code for calculating Effective Resistance Matrix for inputs of Laplacian matrix and number of nodes in the graph N.

```
% Program name: resistance.m
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-03-12
%
% Function to Compute effective resistance matrix for a given Laplacian
% matrix and number of nodes of ER graph.
% Output - Effective resistance matrix


function [omega] = resistance(lap,n)
% Initialise psuedo-inverse(Q) to all zeros.
qinv = zeros(n);

% Compute eigen values and eigen vectors of Laplacian
[v,d] = eig(lap);

% Compute pseudo-inverse(Q)
for i = 2:1:n
    z = v(:,i);
    dia = d(i,i);
    qinv = qinv + ( (1/dia) * z * transpose(z));
end

```

```
% All one vector
u = ones(n,1);

% Diagonal elements of pseudo-inverse(Q)
diagonal = diag(qinv);

% Compute effective resistance matrix
omega = u * transpose(diagonal) + diagonal * transpose(u) - 2 * qinv;
end
```

Matlab Code for generating connected unweighted ER graph for the inputs of number of nodes N and link probability p_ER.

```
% Program name: generate_ER.m
% Provided by: Bastian Prasse
% Date created: 2021-03-12
%
% Generates connected unweighted ER graph
% for the input parameters number of nodes N and link probability p_ER
% Output - Adjacency matrix

function [A] = generate_ER(N, p_ER)

L = N*(N-1)/2;
connectedComponents = 2;
while connectedComponents>1
    A = zeros(N);
    A(triu(ones(N), 1)>0) = (rand(L, 1) <= p_ER) ;
    A = (A + transpose(A));
    [connectedComponents, ~] = graphconncomp(sparse(A), 'Directed', false);
end
```

B

Appendix

Matlab Code for computing hopcount distribution of the weighted random graph

```
% Program name: generate_D
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-07-02
%
% Calculates hopcount distribution of the weighted graph
% Inputs - Unweighted/Weighted graph G, number of nodes N


function [dist] = calculate_hopcount(G,N)

% Initialise empty array
H = [];

% Compute hopcount between all the node pairs
for i = 1:1:N
      for j = i+1:1:N
          [P] = shortestpath(G,i,j);
          d = numel(P)-1;
          H(i,j) = d;
      end
end

H = H(:);
dist = zeros(1,N);

% Creates Hopcount distribution
for k = 1:1:numel(H)
  if(H(k) ~= 0)
     dist(H(k)) = dist(H(k)) + 1;
  end
end

end
```

C
Appendix

Matlab Code to generate demand matrices D for a given N, p and r of weighted ER graph.

```
% Program name: generate_D
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-07-02
%
% Generates demand matrices for a given N and p of weighted ER
% graph. The weights of the ER graph are given through average link
% resistance r with minimum resistance rmin and maximum resistance rmax.

% Input to the program
N = 25;        % Number of Nodes
p = 0.25;      % Link density
rmin = 0.1;    % Minimum link resistance
rmax = 40.0;   % Maximum link resistance
rlz = 1000;    % Number of instances of demand matrices
emin = 0.001; % Minimum value of random element 'e' added to 'effective_resistance'
emax = 0.003; % Maximum value of random element 'e' added to 'effective_resistance'

for rlz = 1:1:rlz
    % Generates unweighted ER graph for given N and p
    A = generate_ER(N, p);

    % Generate random link weights with rmin and rmax
    link_weights = rmin + (rmax - rmin).*rand(N,N);
    c = tril(A,-1);
    weight_tril = c .* link_weights;
    weight_adj = weight_tril + transpose(weight_tril);

    % Round the weighted adjacency matrix to two decimal digits
    weight_adj = round(weight_adj,2);

    % Construct flow based Adjacency matrix
    flow_adj = 1./weight_adj;
    flow_adj(~isfinite(flow_adj)) = 0;

    % Construct degree vector
    val = sum(flow_adj,2);

    % Construct Laplacian
    lap = diag(val) - flow_adj;

```

```
    % Construct effective resistance matrix
    effective_resistance = resistance(lap,N);

    % Construct a matrix of random elements
    e = emin + (emax - emin).*rand(N,N);
    e = tril(e,-1);

    Adapted_effective_resistance = effective_resistance + e;

    % Generate Demand matrix D
    D =  (N/sqrt(2)).* sqrt(Adapted_effective_resistance);

    % convert D to 1D array to store in an excel
    D_array = D(:);

    % Matrix to store the arrays to excel
    matrix_store(:,rlz) = D_array;
end

% Uncomment below line to store the data to desired excel
% writematrix(matrix_store,'/home/sdarsi/bulk/exceldata/er_graph/'+ str2+ '.xlsx');
```

Matlab Code to implement ISPA

```
% Program name: ISPA_implementation
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-07-02
%
% Implements ISPA

% Uncomment this line if input is from an excel
% matrix_store = xlsread('Demand_matrices.xlsx');

% Check to see if negative weighted Adjacency is obtained
check1 = 0;
% Check to see if non-negative weighted Adjacency is obtained
check2 = 0;
% Check to see if S matrix is less than demand matrix D.
success = 0;

for rlz = 1:1000
    % Get a instance of demand matrix D (1D array)
    D = matrix_store(:,rlz);

    % Convert 1D array to 2D array
    D = reshape(D,N,N);

    % Check if D is a square matrix and determine the number of nodes
    [row, column] = size(D);
    if row == column
      disp('D is a square matrix.Proceed');
      N = row;
    else
      disp('D is not a square matrix. Correct it');
    end

    % ISPA line 1
    omega = (2/N^2).* D .* D;


    %% Conversion of D to distance matrix

    % Check if all the elements of D are greater than or equal to 0 and then
    % proceed

    if all(omega(:)>= 0)
```

```
        disp('Distance matrix contains non-negative elements.So proceed');

        %Check to see if D contains zero diagonal elements
        if all(diag(omega) == 0)
            disp('Distance matrix  contain zero diagonal elements. So it is right. Hurray!');
            disp('Checking if distance matrix is symmetric');
            if issymmetric(omega)
                disp('Yes it is symmetric. Proceed!');
            else
                disp('Distance matrix is not symmetric. Make it symmetric');
                for i = 1:1:N
                    for j = i+1:1:N
                        if omega(i,j) ~= omega(j,i)
                            min_value =  min(omega(i,j),omega(j,i));
                            omega(i,j) = min_value;
                            omega(j,i) = min_value;
                        end
                    end
                end

                disp('Now Distance matrix should be symmetric. One step ahead');
                %disp(omega);
            end

            %Check if triangular inequality holds for each element of D and if not
            %change the elements accordingly.
            for i = 1:1:N
                for j = i+1:1:N
                    for k = 1:1:N
                       if i ~= k && k ~= j
                         if omega(i,k) + omega(k,j) < omega(i,j)
                              for x = 1:1:N
                                  min_replace(x) = omega(i,x) + omega(x,j);
                              end
                              min_value = min(min_replace);
                              omega(i,j) = min_value;
                              omega(j,i) = min_value;

                          end
                       end
                    end
                end
            end
            disp('Now triangular inequality also holds. So distance matrix is');
            %disp(omega);
        end

    else
        disp('Distance matrix D contains negative elements.Correct it');
    end

    %% ISPA implementation %%


    inv_omega = inv(omega);

    %All one vector
    u = ones(N,1);

    term1 = u' * inv_omega * u;
    p = (1/term1) * inv_omega * u;
    var = 1/(term1*2);

    % Laplacian from ISPA
    Q = (1/var )*(p * p') - 2 * inv_omega;

```

```
    % Weighted Adjacency of flow networks
    A_calculated = -(Q - diag(diag(Q)));
    A_calculated = round(A_calculated,4);

    % Weighted adjacency of path networks
    A_reversed = 1./A_calculated;
    A_reversed(~isfinite(A_reversed)) = 0;


    g1 = graph(A_reversed);

    % Check if A is non-negative
    if any(A_reversed(:) < 0)
        check1 = check1 + 1;
    else
        check2 = check2 + 1;

        % Feasibility check of ISPP lower bound
        [feasibility_pass_belowr0,feasibility_fail_belowr0,
            feasibility_pass_abover0,...
            feasibility_fail_abover0 ] = feasibility_check(A_reversed);

        % Shortest path matrix
        s1 = distances(g1);

        % Check to see if S <= D
        if all(s1(:) <= D(:))
            success = success + 1;
        end
    end

end
```

Matlab Code which checks feasibility of ISPP bounds for the solutions obtained through ISPA

```
% Program name: feasibility_check
% Author: Sai Poojitha Darsi (s.p.darsi@student.tudelft.nl)
% Date created: 2021-07-02
%
% Checks feasibility of ISPP bounds for the solutions obtained through ISPA

function [feasibility_pass_belowr0,feasibility_fail_belowr0,
    feasibility_pass_abover0,...
            feasibility_fail_abover0 ] = feasibility_check(A)

    %Initialization to 0
    feasibility_pass_belowr0 = 0;
    feasibility_fail_belowr0 = 0;
    feasibility_pass_abover0 = 0;
    feasibility_fail_abover0 = 0;

    % Obtain number of nodes N if A is a square matrix
    [row, column] = size(A);
    if row == column
      disp('A is a square matrix.Proceed');
      N = row;
    else
      disp('A is not a square matrix. Wrong data');
    end

    %Compute number of links
    links = sum(A(:) > 0)/2;

    %Compute link density
    link_density = 2*links/(N*(N-1));
```

```

    %Compute average link resistance r
    r = sum(A(:))/(2*links);
    r0 = 0;

    %Compute critical threshold r0
    if 0.05 <= link_density && link_density <= 0.1
        r0 = 10^(-0.859) * (N^0.812);
    elseif 0.15 <= link_density && link_density <= 0.35
        r0 = 10^(-1.047) * (N^1.089);
    elseif 0.45 <= link_density && link_density <= 0.55
        r0 = 10^(-0.963) * (N^1.178);
    elseif 0.70 <= link_density && link_density <= 0.80
        r0 = 10^(-0.879) * (N^1.215);
    elseif 0.95 <= link_density && link_density <= 1.0
        r0 = 10^(-0.797) * (N^1.231);
    end

    %Flow adjacency
    A_F = 1./A;
    A_F(~isfinite(A_F)) = 0;

    % Valency
    val = sum(A_F,2);

    % Laplacian
    lap = diag(val) - A_F;

    %Effective resistance matrix
    omega = resistance(lap,N);

    %Shortest path matrix s
    g = graph(A);
    s = distances(g);

    %m
    inv_link_wt = 1./A;
    inv_link_wt(~isfinite(inv_link_wt)) = 0;
    m = sum(inv_link_wt,'all')/2;

    %Q-norm
    diffX = omega - ((1/m).*s.*s);

    % Feasibility check
    if r <= r0
        if diffX >= 0
            feasibility_pass_belowr0 = feasibility_pass_belowr0 + 1;
        else
            feasibility_fail_belowr0 =  feasibility_fail_belowr0 + 1;
        end
    else
        if diffX >= 0
            feasibility_pass_abover0 = feasibility_pass_abover0 + 1;
        else
            feasibility_fail_abover0 = feasibility_fail_abover0 + 1;
        end
    end
end
```