Queen Mary University of London

Simplifying large-scale communication networks with weights and cycles

Liu, Ling

Simplifying Large-scale Communication Networks with Weights and Cycles

by

Ling Liu

A thesis submitted to the University of London for the degree of
Doctor of Philosophy

School of Electronic Engineering and Computer Sciences
Queen Mary, University of London
United Kingdom

April 2010

Declaration

The work presented in the thesis is the authors own.

DATE:. . . . .

SIGNATURE:. . . . .

TO MY FAMILY

Abstract

A communication network is a complex network designed to transfer information from a source to a destination. One of the most important property in a communication network is the existence of alternative routes between a source and destination node. The robustness and resilience of a network are related to its path diversity (alternative routes). Describing all the components and interactions of a large communication network is not feasible. In this thesis we develop a new method, the *deforestation* algorithm, to simplify very large networks, and we called the simplified network the *skeleton* network. The method is general. It conserves the number of alternative paths between all the sources and destinations when doing the simplification and also it takes into consideration the properties of the nodes, and the links (capacity and direction).

When simplifying very large networks, the skeleton networks can also be large, so it is desirable to split the skeleton network into different communities. In the thesis we introduce a community-detection method which works fast and efficient for the skeleton networks. Other property that can be easily extracted from the skeleton network is the cycle basis, which can suffice in describing the cycle structure of complex network.

We have tested our algorithms on the Autonomous System (AS) level and Internet Protocol address (IPA) level of the Internet. And we also show that *deforestation* algorithm can be extended to take into consideration of traffic directions and traffic demand matrix when simplifying medium-scale networks.

Commonly, the structure of large complex networks is characterised using statistical measures. These measures can give a good description of the network connectivity but they do not provide a practical way to explore the interaction between the dynamical process and network connectivity. The methods presented in this thesis are a first step to address this practical problem.

Acknowledgments

First and foremost, to my supervisor, Dr Raúl Mondragón, for his great guidance, support and help through every step of my research. During the third year, I have experienced the hard time of the PhD research and lost my enthusiasm for some time. He always patiently talked with me, encouraged me to get out of the bad situation, and pushed me back to the way. I'm really appreciated for his kindly help. And also, I was quite benefited from his ideas and suggestions given for the next step of my future research.

My thanks also to the people working in the Network Research Group on the fourth floor, Vindya, Frank Yang, Tony Yang, etc, for their giving me a quite comfortable and discussional environment when doing the project together, and providing me with the valuable documentation and feasible advice needed as well.

Last but certainly, great gratitude to my parents for their patience and understanding during the tough time despite my moody temper. You all did believe in me and I really thank you for that.

Table of Contents

List of Figures

List of Tables

Glossary

AS	Autonomous system
B	Betweenness Centrality
BGP	Border Gateway Protocol
C^g	Clustering coefficient (g = the length of cycles)
M	Cyclomatic number
DFS	Depth First Search
ECMP	Equal Cost Multi Path
G	An undirected graph
$g_{s,d}$	The number of shortest-path from source s to destination d.
IGP	Interior Gateway Protocol
IPA	Internet Protocol Address
IS-IS	Intermediate system to intermediate system
ITDK	Internet Topology Data Kit
k	The degree of nodes
L	A finite set of links
LAN	Local Area Network
LIFO	Last In First Out
MST	Minimum Spanning Tree
N	A finite set of Nodes
OSPF	Open shortest path first

$\Phi(r), \phi(r)$	Rich-club coefficient
Q	Modularity
QoS	The Quality of Service
STP	Spanning Tree Protocol
TCP	Transimission Control Protocol
W	Weight
Z_m	Cycle space

Chapter 1

Introduction

1.1 Research motivation and objectives

A communications network is a complex network designed to transfer information from a source to a destination. As the information travels through the network it passes through a large number of heterogeneous devices. Today's Internet is a well-known example, which is a large-scale, highly engineered, yet highly complex communications network [60]. It has experienced an explosive development and continues to undergo significant changes over time. In terms of size, by mid-2009, the Internet consisted of about 1.67 billion hosts or endpoints, and more than 100,000 distinct networks, totalling hundreds of millions of routers and links connecting the hosts to the routers and the routers to one another [64, 79]. In the UK [63], in 2008 there were 7.5 million internet banking customers, 40% of business transactions were on-line in 2008, and NHS and social services targeted a 50% increase in the number of people using community equipment services (including tele-care remote monitoring of symptoms). Thus there are substantial economic and social benefits to be gained by improving both our understanding of structure and behaviour of the Internet and its properties. Accordingly to the growth of network size, a substantial movement of focus in network research was shifted away from the analysis of single small networks or properties of individual nodes to consideration of large-scale statistical properties of networks. Therefore, the availability of computers and the data that allow us to gather from communication networks are on a scale far larger than previously possible. This change of scale then forces upon us a corresponding change in our analysing and modelling approach.

Companied with the network's size increasing, the likelihood of one or more network elements failing also increases. Consequently, network reliability becomes an important issue in the design of large-scale complex networks. Generally, the connection between two nodes in a network is considered *resilient* if there are at least two independent paths between them, so that the failure of a single path would not cause the network to be disconnected. The number of alternative paths in the network gives an idea of the network robustness to failure. Understanding the resilience of the network can be used to assist adaptive routing if a failure occurs, optimise the network performance under unusual conditions, and predict or avoid the congestion.

The aim of our research primarily concentrates on simplifying large scale network structure while keeping the network's resilience, characterising the large communications network topological features and exploring the interaction between topological features and network dynamics.

Simulation plays a vital role in studying the complex behaviour of both existing small to medium size communications networks, and proposed future architectures. Using simulation to study Internet–scale networks is not presently feasible [66] and is not likely that it would be [59]. The challenge that a researcher confronts when developing or testing new Internet services is to test a concept designed to work on networks of tens of millions of elements using simulations of only few thousand elements. There exist some methods to simulate networks with tens of thousands of connections. The simulators are based on parallel computing or in some abstraction of the simulation process. For example the *straw-man approach* [66], where the concept/protocol to test is first modelled in a small network. The next step is to increase the size of the network and look for invariants in the behaviour of interest. Other possible approach is *selective abstraction* [34] of the Internet elements and dynamics, such that, the results from a computationally feasible network simulation can be extrapolated to

Internet size networks.

1.2 Research contributions

Based on selective abstraction, we have developed new algorithms to simplify very large networks. The simplified network has the same path diversity as the original network. Path diversity is related to the existence of many different routes, which in turn is associated with the robustness of the network to failures. Thus the new algorithm conserves the number of alternative paths between all the source and destinations when simplifying. The algorithms are general and they have also been extended to take into consideration the properties of the nodes, the capacity and directions of links [40–43].

The work reported in this thesis is novel, and the main contributions being:

- A novel network simplification algorithm has been implemented in the thesis. It conserves the number of alternative paths when simplifying and it has been done based on graph theory (chapter 3).

- The connectivity of the simplified network is not unique when the algorithm starts from different initial nodes. This could be an advantage point that we may construct a simplified network that satisfies given network traffic constraints, and we explore this to extend our simplification algorithm (chapter 3).

- Extended network simplification algorithms have been done, taking into consideration the properties of the nodes, the capacity and direction of links. For example, all the links of the network are weighted and a condition is imposed to simplify the network following the importance of the links, thus the order of topology contraction is controlled, such as the least important links are contracted first, then the more important links, and leave the most important links till the end. This extended algorithm also works in directed and undirected networks (chapter 5, 6 and 7).

- There are several methods to reveal the community structure in networks. The accuracy of these methods depend on the special properties of the network. By studying the properties of the simplified network we are able to choose a community-detection algorithm that is fast and accurate. Using this method, the simplified network can be further decomposed via its community structure (chapter 8).

- The simplified network consists of only cycles, the majority are short cycles, from which the cycle basis can be extracted. The cycle basis is the set of cycles that could fully describe the cyclical structure of the network, and this basis has a wide range of applications. In this thesis a heuristic cycle basis algorithm has been implemented, and possible future work using this cyclical structure is discussed (chapter 9 and 10).

1.3 Organisation of the thesis

In chapter 2 we introduce the concepts behind our approach, including topological description of a network, layouts of global Internet and a brief summary of challenges for modelling the Internet.

Chapter 3 describes the algorithm used to simplify the network and an analysis of the basic features of the simplified network is included. Later in chapter 4 we present examples of the simplification algorithm when applying to the Internet topology.

Chapter 5 shows the simplification algorithm can be extended to take into consideration network dynamics. This is done by guiding the order of the network simplification according to the weights of links.

Comparing with chapter 5, chapter 6 and 7 expands the algorithm with the traffic directions and traffic demand matrix, and shows how the estimation of traffic dynamics affect the order of network simplification.

To summarise, from chapter 3 to chapter 7, the thesis illustrates the novel algorithm to simplify networks by topology only, with a consideration of approximated link utilisation, and with constraints of traffic directions and demands.

In chapter 8, we take a further step to simplify the network by decomposing the simplified network into communities. Chapter 9 investigates the cyclic structure of the simplified networks.

All the work in this thesis is reviewed and concluded in chapter 10, also suggestions on how to extend the research are presented in this chapter.

Chapter 2

Background

Our research primarily concentrates on characterising the large communications network topological features, simplifying large network structure while keeping the network's resilience and exploring the interaction between topological features and network dynamics. We believe the way that the elements of the network are connected to each other have an impact on the network functionality. In this chapter we introduce the basic concepts behind our research.

The communication networks can be described where the hosts, routers and switches are represented by *nodes* and physical connections between them are represented by directed or undirected *links*. A node can transfer information to another node in the form of data packets if there is a link between them. If there is no direct link between the nodes, then a *path* in the network is the sequence of distinct nodes visited when transferring data packets from one node to another. We consider networks where exists at least one path connecting any pair of nodes of the network. The links can have a direction. In this thesis we always consider undirected links at first; and then we would examine if the concepts, algorithms and statistical measurements can apply to directed networks.

2.1 Topological description of a network

2.1.1 Degree distribution

A network is normally described as an undirected graph $G = (N, L)$ where $N = \{n_1, n_2, \cdots, n_i, \cdots, n_N,\ i \in N\}$ is a finite set of N nodes and $L = \{l_1, l_2, \cdots, l_i, \cdots, l_L,\ i \in L\}$ is a finite set of L links. That is N is the number of nodes in the network and L is the number of links. Two nodes are *neighbours* if there is a link joining them. The number of links pointing to a node, k, is known as the *degree* of the node, whose distribution gives the network *connectivity*. The connectivity of the nodes can be described by the *adjacency matrix*. An adjacency matrix is a means of representing which nodes of a network are adjacent to which other nodes. Specifically, the adjacency matrix of a network $G = (N, L)$ is the $N * N$ matrix as shown below.

$$Adjacency\ Matrix = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{iN} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{Nj} & \cdots & a_{NN} \end{pmatrix}$$

whose entry $a_{ij(i \neq j)} = 1$ if node n_i is adjacent to node n_j and 0 otherwise. For undirected graphs $a_{ij} = a_{ji}$. The degree k of a node is the number of neighbours that a node has, $k_i = \sum_j a_{ij}$. The degree is the principal parameter to characterise a node. Two of the simplest properties of a network are its maximum degree $k_{max} = max\{k_i\}$, $i = 1, \cdots, N$ and its average degree $\bar{k} = \frac{\sum_{j=1}^{N} k_j}{N}$.

The spread of nodes degree in the network can be characterised by the node degree distribution $p(k, n_i, N)$, which is the probability that certain node n_i in the network with N nodes has k connections. A plot of node degree distribution for any given network can be formed by making a histogram of the nodes degree. The total degree distribution is

$$P(k, N) = \frac{1}{N}\left(\sum_{i=1}^{N} p(k, n_i, N)\right). \tag{2.1}$$

From the degree distribution, it is easy to obtain another way of calculating the mean degree for a network (average degree): $\bar{k} = \sum_k kP(k)$.

Similarly, in directed networks, there are in-degree k_{in} and out-degree k_{out} for an individual node, hence the network has in-degree distribution $p_{in}(k_{in}, n_i, N)$ and out-degree distribution $p_{out}(k_{out}, n_i, N)$ [20].

The degree of a node measures the number of nearest neighbours of a node (directly connected). It is a local quantity. However, the node degree distribution of the entire network gives important information about the global properties of a network and can be used to characterise different network topologies. In a regular-symmetric network, the degree of every node is the same, making the degree distribution a constant value. In a random network of the type studied by Erdős and Rónyi [21, 22], each link of the network is present or absent with equal probability p, and hence the degree distribution forms a Poisson distribution $P(k) \simeq e^{-pN}\frac{(pN)^k}{k!}$ with a peak at $P(\bar{k})$. As a function of k, the Poisson distribution can be derived as a limiting case of the binomial distribution $P(k) \simeq e^{-pN}\frac{(pN)^k}{k!} = e^{-<k>}\frac{<k>^k}{k!}$ $(N \rightarrow \infty)$. Real-world networks are mostly found to be very unlike random networks in their degree distributions. [23] shows that the majority of the nodes have few neighbours and there is a small set of nodes that have a very large number of neighbours. Those networks exhibit a power law decay in their degree distribution $P(k) \simeq k^{-\gamma}$ where γ is a constant whose value is typically in the range $2 < \gamma < 3$ for real networks. Sometimes this distribution is also called *heavy-tailed distribution.* The heavy-tailed distribution is probability distributions whose tails are not exponentially bounded, that is, the distribution have heavier tails rather than the exponential distribution [1]. Power-law distribution, like many other distributions with heavy tails, is treated as one of heavy-tailed distributions, which describes the same degree distribution from another point of view. A network whose degree distribution follows a power law distribution is known

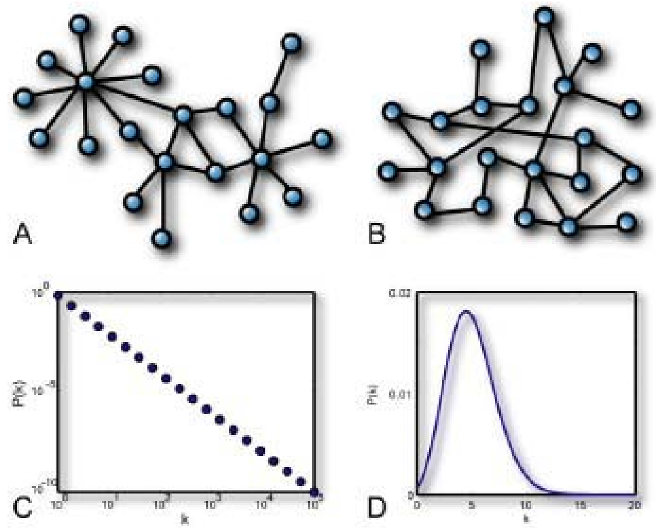


Figure 2.1: (A) Example of a scale-free network. (B) Example of a random network. (C) Scale-free network scales in a power-law degree distribution. (D) Random network demonstrates a poisson degree distribution [17].

as *scale-free* network. This kind of decay is widely present in the technological, biological and sociological networks. Typical examples of scale-free network and random network are shown in figure 2.1, together with the curves of their degree distribution.

2.1.2 Degree correlations

The degree distribution gives only partial information about the network structure. Some questions about relations between node degree have been asked like: do the high-degree nodes in a network associate preferentially with other high-degree nodes? Or do they prefer to attach to low-degree ones?

Assortativity and dissortativity. It has been found that for many real-world networks the degrees of the nodes at either end of an link are not independent, but correlated with one another, either positively or negatively [50, 57, 58]. A network in which the degrees of adjacent nodes are positively correlated is said to show assortative mixing by degree (high-degree nodes preferentially connect to high-degree nodes), whereas a network in which they are negatively correlated is said to show disassortative mixing (high-degree nodes tend to connect to low-degree nodes). Both situations are seen in the networks. It is observed that most social networks appear to be assortatively mixed, whereas most technological and biological networks appear to be disassortative [50].

2.1.3 Transitivity or clustering

In many networks it is commonly found that if node A is connected to node B, node B connects to node C, and node A also connects to node C. In the language of social networks, the friend of your friend is likely also to be your friend. In terms of network topology, transitivity means the presence of a heightened number of triangles in the network - sets of three nodes each of which is connected to each of the others. It can be quantified by defining a clustering coefficient $C^{(3)}$ thus:

$$C^{(3)} = \frac{3N_\triangle}{N_3}, \tag{2.2}$$

where $N_\triangle$ is the number of triangles in the network and N_3 is the number of connected triples. A triangle is a set of three nodes with links between each pair of nodes; a connected triple is a set of three nodes where each node can be reached from each other (directly or indirectly), i.e. two nodes must be adjacent to another node (the central node), shown in figure 2.2. In effect, the clustering coefficient $C^{(3)}$ measures the fraction of triples that have their third link connected such that they form a triangle. The number of triangles and triples are given by

$$N_\triangle = \sum_{k>j>i} a_{ij}a_{ik}a_{jk}, \tag{2.3}$$

$$N_3 = \sum_{k>j>i} (a_{ij}a_{ik} + a_{ji}a_{jk} + a_{ki}a_{kj}), \tag{2.4}$$

where the a_{ij} are the elements of the *adjacency matrix* and the sum is taken over all triples of distinct nodes i, j, and k only once.

An alternative definition of the clustering coefficient, also widely used, has been given by

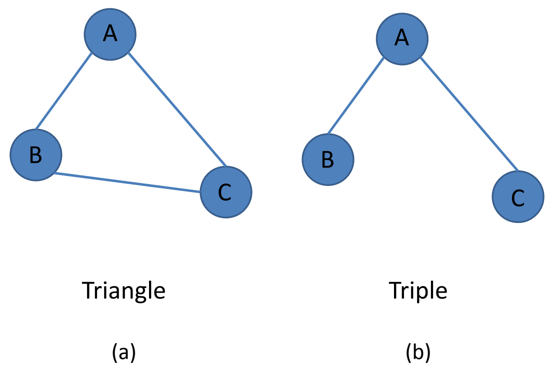


Figure 2.2: (a) A triangle. (b) A triple.

Watts and Strogatz [77], who proposed defining a local value

$$C_i^{(3)} = \frac{N_\triangle(i)}{N_3(i)}. \tag{2.5}$$

For nodes with degree 1, for which both numerator and denominator are zero, we put $C_i^{(3)} = 0$. Then the clustering coefficient for the whole network is the average

$$C^{(3)} = \frac{1}{n}\sum_i C_i^{(3)}. \tag{2.6}$$

Eq.(2.6) calculates the mean of the ratio, rather than the ratio of the means in Eq.(2.2). Eq.(2.6) tends to weight the contributions of low-degree nodes more heavily than using Eq.(2.2) where such nodes have a small denominator, and hence the two equations can give quite different results.

Newman has measured the property of clustering coefficient for several types of published networks [52], shown in table $2-1$. From the table, most of networks have low clustering coefficient, except for the train routes network. This property is very important and it will be used in the following chapter (Chapter 8).

Link clustering coefficient. The link clustering coefficient is defined in analogy with the node clustering coefficient; it is the number of triangles to which a given link belongs,

Network	**type**	n	l	$C_1^{(3)}$	$C_2^{(3)}$
Internet (Autonomous System level)	undirected	11×10^3	32×10^3	0.035	0.39
email messages	directed	60×10^3	86.3×10^3		0.16
email address books	directed	16.9×10^3	57×10^3	0.17	0.13
WWW nd.edu	directed	270×10^3	$1,497 \times 10^3$	0.11	0.29
peer-to-peer network	undirected	880	$1,296$	0.012	0.011
electronic circuits	undirected	24×10^3	53×10^3	0.010	0.030
train routes	undirected	587	19.6×10^3		0.69
power grid	undirected	$4,941$	$6,594$	0.10	0.080
student relationships	undirected	573	477	0.005	0.001
metabolic network	undirected	765	$3,686$	0.090	0.67
marine food web	directed	135	598	0.16	0.23
neural network	directed	307	$2,359$	0.18	0.28

Table 2-1: Basic statistics for a number of published networks. The purpose of this table is to compare two definition of clustering coefficient and also to show the value of clustering coefficient for real networks. The properties measured are: type of network graph, directed or undirected; total number of nodes n; total number of links l; clustering coefficient $C_1^{(3)}$ from Eq.(2.2); and clustering coefficient $C_2^{(3)}$ from Eq.(2.6). Blank entries indicate unavailable data. This table is cited from [52].

divided by the number of triangles that might potentially include it, given the degrees of the adjacent nodes. More formally, for the link connecting node i to node j, the link clustering coefficient is

$$C_{i,j}^{(3)} = \frac{N_{\triangle}(i,j) + 1}{min[(k_i - 1), (k_j - 1)]}, \tag{2.7}$$

where $N_{\triangle}(i,j)$ is the number of triangles that share the (i,j) link and $min[(k_i - 1), (k_j - 1)]$ is the maximal possible number of triangles that can be shared by the link.

Regardless of which definition of the clustering coefficient is used, the clustering coefficient measures the density of triangles in a network, and how closely the nodes connect to their neighbours.

2.1.4 Rich-Club coefficient

There exists another phenomenon in a network; a small number of nodes have large numbers of links, called *rich nodes*. In some networks, these nodes are well interconnected between themselves, forming a *Rich-Club*, and their connectivity tend to dominate the organisation of the network structure.

The connectivity between the nodes belonging to the rich-club can be characterised by *the rich-club coefficient*. Nodes in the network are sorted by decreasing degree. The rank r of a node is its position in the list, i.e. the best-connected is ranked $r = 1$, the second best-connected node $r = 2$ and so on. Rich nodes can be defined as nodes with large degrees or small ranks. The density of connections between the r richest nodes is quantified by the rich-club coefficient [83]

$$\Phi(r) = \frac{2E_{\leq r}}{r(r-1)}, \tag{2.8}$$

where $E_{\leq r}$ is the number of links between the r nodes and $r(r-1)/2$ is the maximum number of links that these nodes can share. If $\Phi(r) = 0$ the nodes do not share any link at all, if $\Phi(r) = 1$ the nodes form a fully connected sub-graph, a clique. As a function of degree, the rich-club coefficient can also be given as [15]

$$\phi(k) = \frac{2E_{\geq k}}{N_{\geq k}(N_{\geq k}-1)}, \tag{2.9}$$

where $N_{\geq k}$ is the number of nodes with degree greater or equal to k and $E_{\geq k}$ is the number of links between the $N_{\geq k}$ nodes. Recently there has been a considerable effort to characterise and model the rich-club connectivity in a variety of complex networks [49, 83]. It is noticed that two networks can have the same $\phi(k)$ and the same degree distribution $P(k)$ for all k, but different $\Phi(r)$ [49]. In the network, having a well connected rich-club means that there is a large number of alternative routing paths between the rich-club members(their average path length is very small, 1 or 2 hops). Hence the rich-club acts as a super traffic hub and provides a large selection of short-cuts. The connectivity between rich nodes can be crucial for network properties, such as network routing efficiency, redundancy and robustness.

2.1.5 Betweenness centrality based on shortest path

In a network, a node can transfer information to another node in the form of data packets if there is a link between them. If there is no direct link between the nodes, then a *path* in the network is the sequence of distinct nodes visited when transferring data packets from one node to another. We consider networks where exists at least one path connecting any pair of nodes of the network, especially in cases where data packet flow primarily follows the shortest available path. The journey time for a packet to go through two shortest-paths with the same length can be very different, due to different traffic patterns and the usage of the routes (paths). Packets can be delayed on the path as they can be accumulated in the nodes' buffers (queues). Therefore, in the network, there are nodes that are much more busier or highly utilised to transfer packets, while other nodes spend most of the time idle. From the network's topological properties, it is possible to approximate the traffic load on the nodes and the bandwidth utilisation of the links. The betweenness centrality measures the "importance" of the nodes and can be used to approximate the loads on the nodes.

Given a source s, and a destination d, the number of shortest-path between them is $g_{s,d}$. The number of shortest-path that contains the node ν is $g_{s,d}(\nu)$. Then the proportion of shortest-paths, from s to d, which contain node ν is $p_{s,d}(\nu) = \frac{g_{s,d}(\nu)}{g_{s,d}}$. The proportion of shortest-paths and shortest-path length are related by $\ell_{s,d} = \sum_{\nu \in V} p_{s,d}(\nu) - 1$, where V is the set of nodes that contains the nodes visited by the shortest-paths from s to d. The definition of betweenness centrality of a node $B(\nu)$ given by [26]

$$B(\nu) = \sum_{s \neq d \in N} p_{s,d}(\nu) = \sum_{s \neq d \in N} \frac{g_{s,d}(\nu)}{g_{s,d}}, \tag{2.10}$$

where N is the set of nodes in the network. A fast algorithm to calculate the betweenness centrality in large-scale networks has been developed by Brandes [9].

Link betweenness centrality. Similar to the node betweenness centrality Eq.(2.10), the link betweenness centrality is calculated as the proportion of shortest paths that travel through an specific link.

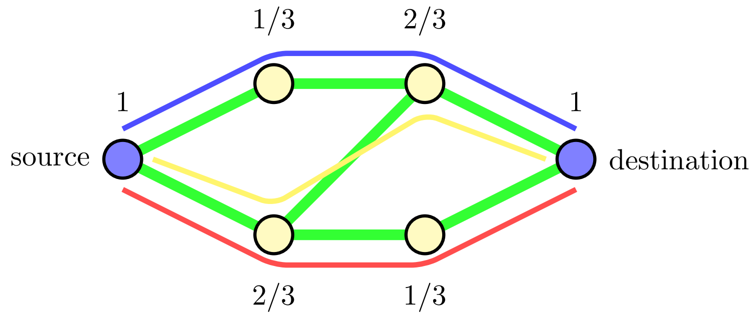


Figure 2.3: Traffic splitting for source s to destination d according to Equal Cost Multi-path (ECMP) rule.

Figure 2.3 shows a simple network, which has three shortest paths between source s and destination d accordingly to Equal Cost Multi-path (ECMP)[1] rule. The traffic between nodes (s, d) is considered to equally split and distribute among these three paths. We denote this ratio of traffic splitting as the factor $\alpha = 1/g_{s,d}$.

The nodes or links with high betweenness centrality are relatively more important in the network as they are visited by more routes. It is expected that the removal of these nodes will worsen the network performance. In the worst situation the failure of a node with high betweenness centrality could cause severe congestion.

Other centrality measures. Conventionally, the betweenness centrality assumes that traffic traverses the network following shortest paths, however there are other centrality measurements that can be used instead of the *shortest-path betweenness centrality* to describe network traffic properties [25, 51]. Table $2-2$ gives a summary of these measures.

2.1.6 Path diversity and cycles

In the networks, a packet would be re-routed if the traffic flow is congested or blocked on the pre-selected shortest-path. If nodes are removed from a network, the typical length of these shortest paths will increase, and ultimately node pairs will become disconnected

[1]Equal-Cost Multi-Path (ECMP) is a forwarding mechanism for routing packets along multiple paths of equal cost with the goal to achieve almost equally distributed link load sharing. This significantly impacts a router's next-hop (path) decision, and multipath routing can be used in conjunction with most routing protocols.

Ways of transition	**Network dynamics**	**Centrality measurement**
shortest path	traffic property packet traffic	centrality node betweenness link betweenness
paths	Internet name server	closeness degree
Trail	email broadcast virus	closeness degree

Table 2-2: Other centrality measurements

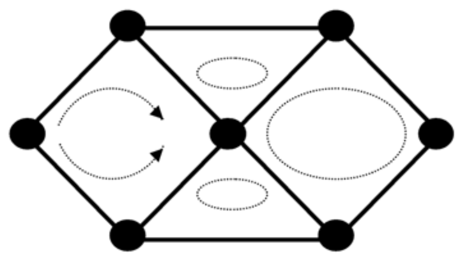


Figure 2.4: An example of path diversity topology. In the network there are more than one alternative path between any pair of paths. If a node, a link or a part of network is congested or broken, the rest of network can still be reach one another. Thus the number of alternative paths gives an idea of the network robustness to failure.

and communications between them through the network will become impossible. Hence the number of alternative paths in the network gives an idea of the network resilience to failure.

Path diversity describes the number of disjoint paths between pair of nodes. The simplest and smallest path diversity unit is a *cycle*, as it provides two path choices to go from any node in the cycle to any other node in the same cycle. The number of alternative paths is strongly related to the existence of cycles. Routing can exploit path diversity to achieve network resilience to congestion and link or node failures. Figure 2.4 shows the alternative paths between each pair of nodes in the network.

Cycles. Cycle structure of graphs is an old topic that has occupied electrical engineers for nearly a century [38]. To obtain more information about the cycle structure of a network recall that the collection of all cycles form a vector space [32, 44], called the *cycle space*. In this space, a cycle is a vector indexed by links, where the i entry of the cycle vector Z_m is one if the link belongs to the cycle Z_m and zero otherwise. A cycle basis of the graph G is defined as a basis for the cycle space. Any cycle Z can be expressed as $\sum_{i=1}^{M(G)} Z_i$ where $Z_1, Z_2, Z_3, \ldots, Z_{M(G)}$ form the cycle basis. The number of cycles in the cycle basis, also called *cyclomatic number* [44], is

$$M(G) = N - L + K, \tag{2.11}$$

where N is the number of nodes, L is the number of links and K is the number of *connected component*[2] in the networks. In here we consider only connected networks, that is $K = 1$. As can be seen in the example of figure 2.4 , the network totally has 11 nodes and 14 links. Then the cyclomatic number, which is the number of cycles in the graph, is quantitied as follows: $M = N - L + K = 14 - 11 + 1 = 4$.

Intuitively, there are 4 cycles in the graph, which is the same as the result obtained from the calculation.

2.1.7 Community structure

It is widely assumed that most real-world networks show *community structure* [29, 53], the division of network nodes into communities; within the community the connections are dense while the connections are sparse between communities, see figure 2.5. The ability to find and analyse such communities can provide invaluable help in visualising the structure of networks and understanding network's dynamical evolution. It is common experienced that people divide themselves into communities along lines of interest, occupation, age and so forth. In the case of the World Wide Web, for instance, pages related to the same subject are typically organised into communities, so that the identification of these communities can help the task of seeking for information. Similarly, in the case of the communications

[2] In graph theory, a **connected component** of an graph is a subgraph in which any two nodes are connected to each other by paths.

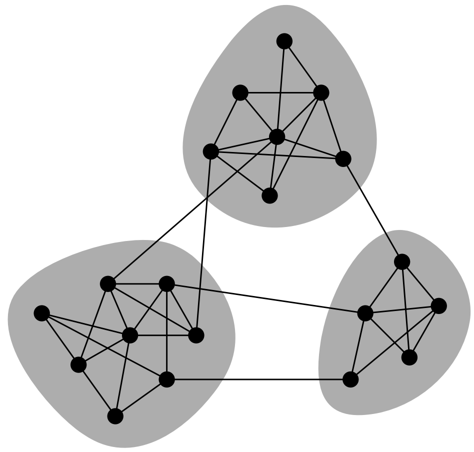


Figure 2.5: The nodes in many networks fall naturally into communities, sets of nodes (shaded) within which there are many links, with only a smaller number of links between nodes of different communities [55].

network, information about communities formed by routers geographically close to one another can be considered in order to improve the flow of data.

Despite the importance of the concept of community, there is no consensus about its definition. An intuitive definition was proposed by Radicchi *et. al.* [61] based on the comparison of the link density among nodes. Communities are defined in a strong and a weak sense. In a strong sense, a subnetwork is a community if all of its nodes have more connections between them than with the rest of the network. In a weak sense, on the other hand, a subnetwork is a community if the sum of all node degrees inside the subnetwork is larger than outside it. One of the consequences is that every union of communities is also a community. This comparative definition is intuitive, and to some extent, describes the search for communities in large complex networks. Several other possible definitions are described in [75].

A wide variety of heuristic algorithms for revealing the community structure have been developed and it is worth reviewing them here. Basically they are classified into agglomerative methods, divisive methods, spectral methods, methods based on optimising the modularity measure [17, 19], which are briefly reviewed as follows.

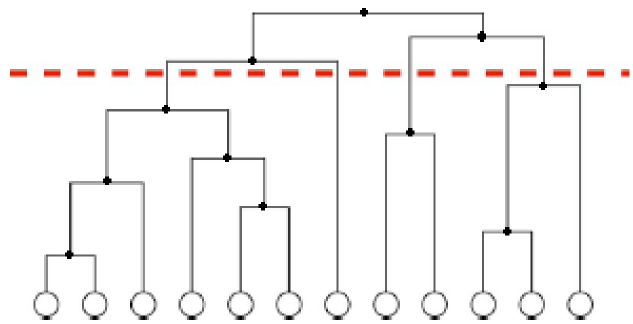


Figure 2.6: A hierarchical tree or dendrogram illustrating the type of output generated by the community algorithms. The circles at bottom represent the individual nodes of the network and the dots at each level represent subnetwork, which can be considered as a community. The dot at the top of the tree represents the whole network.

Agglomerative methods. The traditional method for extracting community structure from a network is *cluster analysis* [75], sometimes also called *hierarchical clustering*. For every pair (i, j) of nodes in the network, one calculates a weight $w_{i,j}$, which measures how closely connected the nodes are. Starting from the set of all nodes and no links, links are iteratively added between pairs of nodes in order of decreasing weight. In this way nodes are grouped into larger and larger communities. Alternatively, the entire progression of the algorithm from the empty graph to complete graph can be represented in the form of a tree or dendrogram such as that shown in figure 2.6. In the figure, as we move up the tree, the node join together to form larger and larger communities, as indicated by the lines, until we reach the top, where all are joined together in a single community. Alternatively the dendrogram depicts an initially connected network splitting into smaller and smaller communities as we go from top to bottom. A cross section of the tree at any level. such as that indicated by the dotted line, will give the communities at that level. The vertical height of the split points in the tree are indicatively only of the order in which the splits (or joins) take place, although it is possible to construct more elaborate dendrograms in which these heights contain other information [56].

Divisive methods. By looking at the problem from a different perspective, we can start with the network as a whole. Intuitively, the simplest way to partition a network is to cut some links until the network is no longer connected, which is the method called *divisive method*, and in which we are interested in our research. This method attempts to find the *least* similar connected pairs of nodes, and then remove the links between them. By doing this repeatedly, we divided the network into smaller and smaller components and again we can stop the process at any stage, and take the components at that stage to be the network communities. Again, the process can be represented as a dendrogram depicting the successive splits of the network into smaller and smaller communities (also see figure 2.6). And the crucial point in a divisive algorithm is the selection of the links to be cut.

Spectral Methods. Spectral methods are based on the analysis of the eigenvectors of matrices derived from the network [70]. These methods have been discussed in a recent survey by Newman [54]. The measurement is related to the eigenvalues of the adjacency matrix minus a probability matrix. The adjacency matrix describe the connectivity between nodes in the network, which is already defined at the beginning of this chapter.

$$a_{ij} = \begin{cases} 1 & \text{if there is a link joining nodes i, j,} \\ 0 & \text{otherwise.} \end{cases}$$

The probability matrix is the expected number of links between nodes i and j, which is calculated as

$$p_{ij} = \frac{k_i k_j}{2L},$$

where k_i, k_j is the degree of the nodes i and j, and L is the number of links of the networks. In order to split the network into communities, the largest eigenvalue is determined. According to the signs of the elements of the eigenvector, the network is divided into two parts, that is, nodes with positive elements are assigned to a community and nodes with negative elements to anther.

Methods based on optimising the modularity measure. An approach that has become widely accepted was proposed by Newman and Girvan in [53], called *modularity*. Modularity is a benefit function used in the analysis of networks such as communication networks or social networks. It quantifies the quality of a division of a network into communities. Good divisions, which have high values of the modularity, are those in which there are dense internal connections between the nodes within communities but only sparse connections between different communities. The most common use of the modularity is as a basis for optimization methods for detecting community structure in networks. Consider a network composed of N nodes connected by L links and let a_{ij} be an element of the adjacency matrix of the network, which gives the number of links between nodes i and j. And suppose a candidate division of network communities is given. The modularity of this division is defined to be the fraction of the links that fall within the given communities minus the expected such fraction if links were distributed at random. Commonly, the randomization of the links is done so as to preserve the degree of each nodes. In this case, the expected number of links falling between two nodes i and j following randomization is $k_i * k_j/2L$, and hence the actual minus expected number of links between the same two nodes is $a_{ij} - k_i * k_j/2L$. Summing over all pairs of nodes in the same community, the modularity, denoted Q, is then given by

$$Q = \frac{1}{2L} \sum_{ij} \left[a_{ij} - \frac{k_i * k_j}{2L} \right] \delta(c_i, c_j), \tag{2.12}$$

where c_i is the community to which node i belongs and $\delta(c_i, c_j)$ is the Kronecker delta[3] symbol. The modularity can be either positive or negative and the value of the modularity lies in the range $[-1,\ 1]$. The positive values indicate the possible presence of community structure. Thus this method has the ability not only to divide networks effectively, but also stops dividing when no good division exists.

Above we review the concepts of the network topological features and their conventional

[3] In mathematics, the Kronecker delta or Kronecker's delta, named after Leopold Kronecker (1823-1891), is a function of two variables, usually integers, which is 1 if they are equal, and 0 otherwise. So, for example, $\delta(1, 2) = 0$, but $\delta(3, 3)$=1.

measurements behind our research. As we are studying the large-scale communications network, i.e. the Internet, it is invaluable to explore the orgnisation of the Internet, its network properties and the challenges for modelling it.

2.2 Layouts of the global Internet

2.2.1 AS level network

For management purposes, the Internet is divided into subnetworks. Each subnetwork adheres to common routing conventions, usually the Interior Gateway Protocol (IGP). The management of a subnetwork and its routers fall under one administrative entity called an Autonomous System (AS). Each AS is a collection of routers and links under a single administrative domain, and the Internet can be considered in an abstract space where the relevant property is the connectivity between ASs. At this level we tend to disregard many physical properties of the network like the geographical location of the ASs, which could be in different continents, or the direction of the links and their capacities.

A coarser view of the Internet can be obtained by aggregating IP address or router into their corresponding ASs. In this way **CAIDA** [10] is providing a visualisation of the AS core by converting each IP address in the AS responsible for its routing. The mapping is made by using the Border Gateway Protocol (BGP) routing tables collected by the *Oregon route-views* project by CAIDA. BGP tables contain the AS paths to destination IP addresses. This data aggregation allows AS connectivity maps to be reconstructed and provides logical layouts that can be used to study the role of specific ASs in routing traffic across the Internet.

Oregon route-views is one of the very few publicly available data sets that allows a dynamical analysis of the time evolution of the Internet, hence it is at the core of many studies of the Internet's AS connectivity structure. With the aim of establishing the completeness of the AS level topology, Qian *et. al.* [13] supplemented and compared these data set with BGP summary information from a number of different sources. This remarkable work provides an *extended* AS graph (*AS*+) of the Internet, which contain 20% − 50% more physical connections, but only 2% more ASs. This finding demonstrates that the graphs obtained from *Oregon route-views* is not complete, and misses a noticeable fraction of the Internet connectivity. Therefore, the extended AS+ map is an essential benchmark from which to test the stability and consistency of statistical measurement of the AS graph.

2.2.2 Router level network

A basic physical description of the Internet should include the geographical position of the nodes and links, the capacity of the links and the direction that the Internet traffic follows. At the router level the nodes and links of the network represent physical entities. The nodes describe the routers and switches that manage the passage of traffic through the network. The links represent the different physical connections between nodes, e.g. optical fibres, coppers, wires etc.. The idea is therefore to find the geographical location of each router or AS, place the node at that very position, and draw lines between physically connected nodes. This strategy sounds simple but is unfortunately very difficult, due to business and security reasons, many ISPs do not want the exact positions of their machines to be publicly available. In many cases it is difficult to exact even an approximate location from the host name, and many routers just have an IP address. Therefore, it s not possible to have an accurate description of the Internet at the route level.

2.2.3 IP address level

Instead, the Internet can be interpreted at the Internet Protocol address (IPA) level. The strategies are to establish a correspondence between IP addresses, domain names, and ASs using the *whois*[4] database, which provides the registered headquarters' address of ISPs.

[4] *Whois* is a query/response protocol that is widely used for querying databases in order to determine the registrant or assignee of Internet resources, such as a domain name, an IP address block, or an autonomous system number. It is publicly available to anyone who chooses to check domain names using the *Whois* search tool. *Whois* services are typically communicated using the Transmission Control Protocol (TCP). Servers listen to requests on the well-known port number 43.

2.3 Features of the global Internet

Here we shall focus on some metrics which provide a basic characterisation of the Internet.

2.3.1 Small-world properties

The average shortest-path length among nodes found in the Internet is very small if compared with the size of the Internet. This observation was reported in early analysis of Internet data [23], and it has been confirmed for all recent data sets. This small separation among Internet routers and ASs is a striking example of the so-called *small-world effect*. This concept has been popularized in the sociological context, where it is sometimes referred as "six degree of separation" [47], later it has been observed in many natural networks [76]. Statistical result reveal that the shortest distance between two random nodes is short, and the average shortest path of the Internet is about 14 hops [13].

2.3.2 Heavy tailed distribution

Recently the study of the Internet's AS topology reported that the node degree distribution of snapshots of the measured AS-connectivity graph follows a power law [23]. This finding implies that, whereas most of the ASs have a node degree of one or two, the probability of encountering a few ASs that are highly connected is significant. This data-driven observation is in sharp contrast to the traditionally theoretical topology models [12, 78], which yield node degree distributions that decay exponentially fast, essentially ruling out the occurrence of high-degree node and giving high probability to "typical" node degrees on the order of the average node degree of the graph. In contrast, no such preferred or "typical" node degree can be identified for power-law node degree distribution and, because of this absence of a characteristic scale, the resulting structures are termed *scale-free* networks.

2.3.3 Rich-club phenomenon

The Internet ASs are very well connected between each other. The statistical result shows the top 1% rich ASs have 32% of the maximum possible number of links, and are connected preferentially to each other. The number of links between the top 5% rich ASs is significantly larger than the numbers of links connecting the rich ASs to other ASs with smaller degree [83].

In [49], the researchers show that two networks can have the same degree distribution and same *small-world* property, but different *rich-club* coefficient. Therefore the rich-club coefficient is very important network property in the network modelling. An Internet model without rich-club phenomenon may under-estimate the efficiency and flexibility of the traffic routing in the AS graph.

As we study the simplification of the large-scale communication networks, among those network properties shown above, some properties are very important and they are kept during the simplification, some properties may be not very important so that they could be sacrificed, and some properties that studied in the simplification may have changes to reveal the underlying network characteristics. For example, in the communication networks, passing the information from a source to a destination all the time is important, therefore, the resilience of networks is interested by our research and retained in the network simplification. Whereas the average shortest path is always reduced after the simplification because the network size is reduce and the distance between any pair of node may be shorter. This parameter (the average shortest path) is already a small value comparing to its size for the original network, so it does not make much difference if it is reduced after the simplification, so that this parameter could be ignored or sacrificed in the research. (For further details, refer to Chapter 4.)

2.4 Challenges of the IP networks modelling

Besides the Internet's features shown above, many simulations and analysis are widely used as an essential tool to explore other metrics of the Internet. However there are several key factors that make the Internet exceedingly hard to characterize and thus to simulate.

First, it is the Internet's size and continued growth. The Internet is growing exponentially and its size has already increased by five orders of magnitude since its birth. State of the art simulations can only accurately simulate relatively small networks, around 0.001% of the size of the Internet. An extremely conservative estimate [66] of Internet behaviour calculated an average of 2.9×10^{11} events per second (assuming a network with $1.1\ times 10^{8}$ hosts, with one router for every 100 hosts). The sheer size of the resources needed to simulate such a large network is beyond existing simulations tools (software and hardware).

Secondly, the Internet is heterogeneous. The Internet's great success is to unify and to seamlessly interoperate diverse networking technologies, which have administered by vastly different policies. While conceptually, the Internet's study uses a unified set of protocols etc, in reality, each protocol (likewise traffic or bandwidth) has been implemented by many different communities and the mix of different applications are used at different sites, often with significantly different features. For example, the bandwidth via different types of links used in the network span a very large range. Some are slow modems, capable of moving only hundreds of bytes per second, while others are state-of-the-art fibre optic links with bandwidths millions of times faster. Some traverse copper or glass wires, while others increasingly, are radio- or infrared-based and hence wireless with much different loss characteristics and sometimes complex link layers. Thus, the specifics of traffic types, bandwidth, and protocols become another challenge to tackle.

Thirdly, the Internet is a self-organising system, whose properties cannot be traced back to any blueprint or chart. It evolves and drastically changes over time according to evolutionary principles dictated by the interplay between cooperation (the network has to work efficiently) and competition (providers wish to earn money). So that routers and links are added by competing entities according to local economic and technical constraints, leading to a very intricate physical structure. In a complex network, *form* and *functionality* are closely related. Therefore, an accurate topological description – the specifics of how the individual nodes in the network are connected (directly or indirectly) with each other, and the properties of the links that foster the interconnection is very important. Since the Internet's topology is constantly changing and not all of the Internet traffic carriers are willing to provide fully topological information, there is agreement in the research community on which properties of the network topological models should be based, or how to test their accuracy. Consequently, the Internet structure is difficult to characterize.

To summarise, modelling the large-scale communication networks behaviour is an immensely challenging undertaking because of its sheer size, complex structure and great heterogeneity. The topology of the Internet may be not very complex, but employing various routing protocols/policies within or outside the Internet Autonomous System is the very complicated. The combination of all of these factors result in a general lack of understanding about the large-scale topological structure and performance properties of the Internet. This poor knowledge of the Internet results in no tools being available to evaluate and forecast growth trends and performance problems. For these reasons, in recent years, many research groups have started to deploy technologies and infrastructures in order to obtain a more global picture of the Internet.

Besides the simulation challenges, network reliability has recently become an important issue in the design of large-scale complex networks. That is because the likelihood of one or more network elements failing constantly increases along with the growth of Internet. In the network, path diversity is given by the number of cycles, and is strongly related to the resilience and robustness of the network. Thus, our research primarily concentrates on simplifying large networks to a size that can be modelled, while keeping the network's path diversity. This supports the studying of the Internet's topological characteristics and the interactions between the topology and dynamics in the abstract network.

2.5 The OSI Model

The Internet can also be interpreted as the Open System Interconnection Reference Model (OSI Reference Model or OSI Model), which is an abstract description for layered communications and computer network protocol design. It was developed as part of the Open Systems Interconnection (OSI) initiative [80]. In its most basic form, it divides network architecture into seven layers which, from top to bottom, are the Application, Presentation, Session, Transport, Network, Data Link, and Physical Layers. It is therefore often referred to as the OSI Seven Layer Model, see figure 2.7.

A layer is a collection of conceptually similar functions that provide services to the layer above it and receives service from the layer below it. For example, a layer that provides error-free communications across a network provides the path needed by applications above it, while it calls the next lower layer to send and receive packets that make up the contents of the path. Conceptually within one layer two individuals are connected by a horizontal protocol connection on that layer. The function of each layer can be found in [80].

In previous section, we introduce the Internet can be interpreted at three different granularity levels: the AS level, the IP-Address level and the router level. They are corresponding to lower layers of the OSI model: network layer, Data link layer and physical layer. These three layer are related, i.e. layer 3 is mapped to layer 2 and layer 2 is mapped to layer 1. Although they have relations, they represent different levels of the network connectivity and could have high degree of independence. The network layer is the logical representation of the Internet, the data link layer is traced following by a data packet through a series of IP addresses, and the physical layer reflects the geographical information of network connectivity. Our novel algorithm is general. It can work for all these three layers and simplifies the network at the each layer. Therefore the network simplification at different levels exhibit different information of the network connectivity. An example of the Internet simplification at two granularity level, the AS level and the IP-Address level, are shown in Chapter 4, and we will do the further discussion in that chapter.

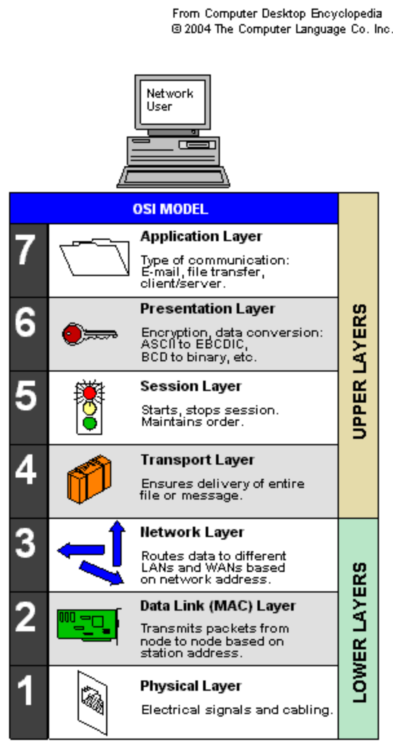


Figure 2.7: Description of the OSI seven layer model [48].

2.6 Summary

The global Internet is a prime example of a large-scale network. As there is a strong interaction between the network's topology and its functionality, it is invaluable to investigate the interesting features arising along with the growth of Internet. Due to its sheer size, complex structure and great heterogeneity, modelling the large-scale communication networks behaviour is an immensely challenging undertaking. Network robustness and resilience recently have become important issues in the design of large-scale complex networks and they are related to its path diversity (alternative routes), thus our research primarily concentrates on simplifying large network structures, while keeping the network's resilience, and also characterising the topological features in their abstract network structure.

Chapter 3

Topology Simplification

3.1 Introduction

In this chapter we are going to develop the techniques to simplify the network's topology. First, we introduce a new technique which could simplify large-scale telecommunication networks. This algorithm simplifies networks by conserving the number of alternative paths, whilst preserving network routing properties. Afterwards, we review some related network simplification algorithms.

3.2 Complexity, resilience and simplification

As the number of elements (nodes and links) in a network increases, the likelihood of failure also increases. Consequently, network reliability becomes an important issue in the design of large-scale complex networks. Network resilience is the ability to resist failure and the adaptivity of routing. Resilience has been studied extensively in communications networks [2, 11, 14]. Generally, the connection between two nodes in a network is considered resilient if there are at least two independent paths between them, so that the failure of a single path would not cause the network to be disconnected. The number of alternative

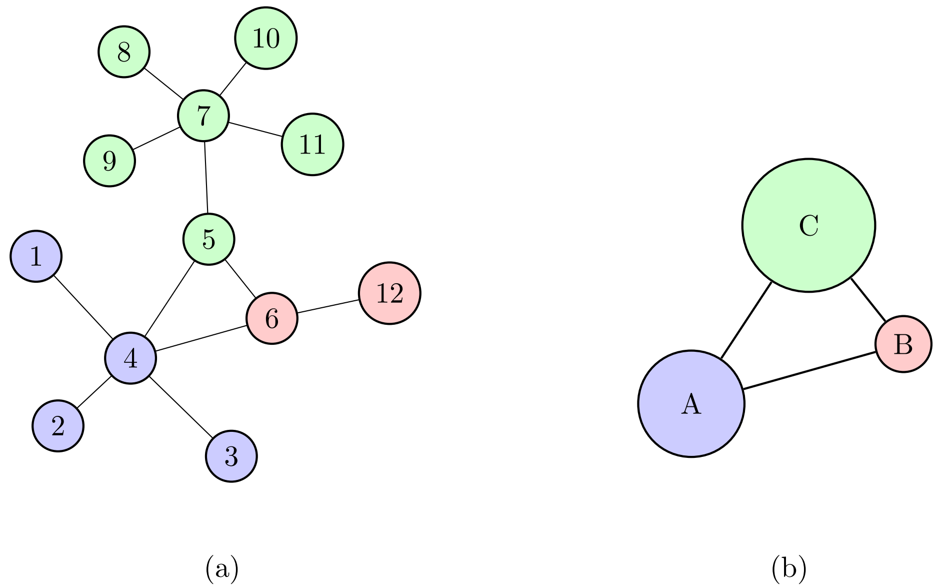


Figure 3.1: Comparison between (a) the original topology and (b) the simplified topology of the network.

paths in the network gives an idea of the network robustness to failure. In the context of communications networks, the multipath structure of a network is used for a number of purposes, such as; increasing fault-tolerance and enhancing reliability, distributing traffic, bandwidth aggregation, and improvement in QoS metrics such as delay.

3.3 Conserving alternative routes

We postulate that a fundamental property for resilience is the number of alternative paths, known as *path diversity*. Without this property there are no routing decisions to be made and adaptivity in routing becomes irrelevant. For example, a *tree* is a graph where all the nodes are connected and there is only one path between any pair of nodes; that is, there are no alternative routes between two nodes in a *tree* [37]. A routing decision on a network whose topology is a *tree* is unique and can be computed trivially [37].

In a network, it is possible to find a subset of the nodes which are all connected and form a tree. If this subset of nodes are grouped together, and represented by a “big” node, the network would be simplified and maintain the same number of alternative paths as the original network. A simple example is in figure 3.1. The nodes in the network can be clustered into three groups, namely $A = \{1, 2, 3, 4\}$, $B = \{6, 12\}$ and $C = \{5, 7, 8, 9, 10, 11\}$. Each cluster consists of a tree, so there is only one route between any pair of nodes belong to the same cluster, i.e. in the cluster C there is one and only one path between node 5 and node 11. However, the path between the nodes which belong to different clusters may not be unique, and all the alternative routes are kept, such as routes from node 3 to node 7. Figure 3.1(b) shows the contracted network which retains the same number of alternative routes between the clusters and it is more intuitive to see the alternative routes.

3.4 Deforestation

The reduction from the original network to a smaller network is done by node contraction. A set of nodes are contracted to a single node if they do not reduce the number of alternative paths in the contracted network. We call the nodes that contain the set of contracted nodes, *super-nodes*. Figure 3.2 shows an example of node contraction. The nodes inside a super-node forms a tree, and the super-node and two adjacent nodes forms a triangle, i.e. the smallest unit of path diversity.

3.4.1 The procedure

We refer to the procedure of contracting all subsets of nodes in the network as *deforestation*. The *deforestation* algorithm is the *depth-first search* (DFS) algorithm [37] and is described as follows:

1. Start from any node.

2. Choose one of its neighbours (go one level down in the search of *trees*).

3. If there is another neighbour; do they form a triangle?

    - Yes:

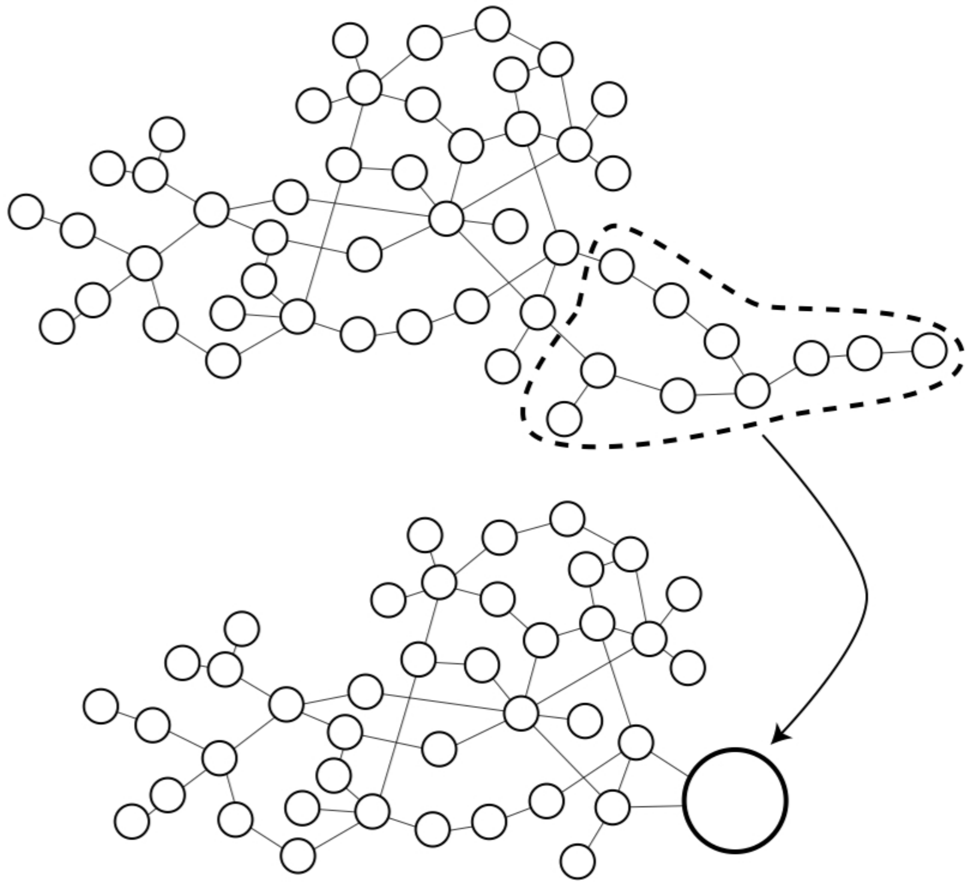


Figure 3.2: Contraction of a set of nodes. The group of nodes inside the dotted line on the upper graph are contracted to a "big" single node at the bottom, because they do not introduce multiple paths when contracted and they form a tree. The "big" node and its two adjacent nodes form the smallest path diversity unit, a triangle. We call the "big" node, the *super-node*. The nodes contained in a super-node form a tree. In the figure, it is not possible to include more nodes in the super-nodes as this would result in the introduction of double links between the nodes or super-nodes.

- If there are more neighbours choose another one. Go to 3.
- If there are no more neighbours go up one level in the depth–search. Go to 2.
- If there are no more neighbours and the depth–search is finished then all the nodes belonging to one super–node have been obtained.

- No: Group the node and its neighbour by giving them the same name (super–node). Go to 2.

The procedure is also shown in the flow chart, see figure 3.3 on the next page.

The algorithm runs recursively. Nodes in trees, along with their corresponding links are collected into super-nodes. We call the contracted network the *skeleton* network.

3.4.2 Implementation of the deforestation algorithm

The implementation of the *deforestation* algorithm uses the representation of the graph as a linked list (see figure 3.4) [37]. Each node has a pointer to its neighbours. If a set of nodes are grouped into a super–node, a new node is created with the linked list to its neighbours. Notice that in the linked list, a super–node can contain a pointer to itself.

The algorithm is recursive (it is possible to implement the algorithm using a LIFO (Last In First Out stack for expansion) based on the depth-first search. The maximum depth searched by the algorithm depends on the number of neighbouring nodes and if the inclusion of a node in the list means we have a triangle in the skeleton network.

The pseudo code of the algorithms are the following. Algorithm 1 checks if any three neighbouring nodes form a triangle or not. More intuitively, we use *mother, daughter, granddaughter* as nodes in the algorithms to represent the triangular relationship. Algorithm 2 describes the procedure of contraction. It starts with the discovery of one of the trees in the original graph. The nodes that belong to that tree are collected in a temporary list (referred as L). The collection of nodes ends when the algorithm runs out of nodes, or when the inclusion of a new node creates a cycle (i.e. a triangle) in the nodes collected in the temporary list. And algorithm 3 give the general procedure of the *deforestation* algorithm.

Algorithm complexity. As the *deforestation* algorithm is based on the DFS algorithm, the total time to perform the *deforestation* is $O(n + l)$, where n is the number of nodes and l is the number of links. For an arbitrary node a, its neighbours and the neighbours of

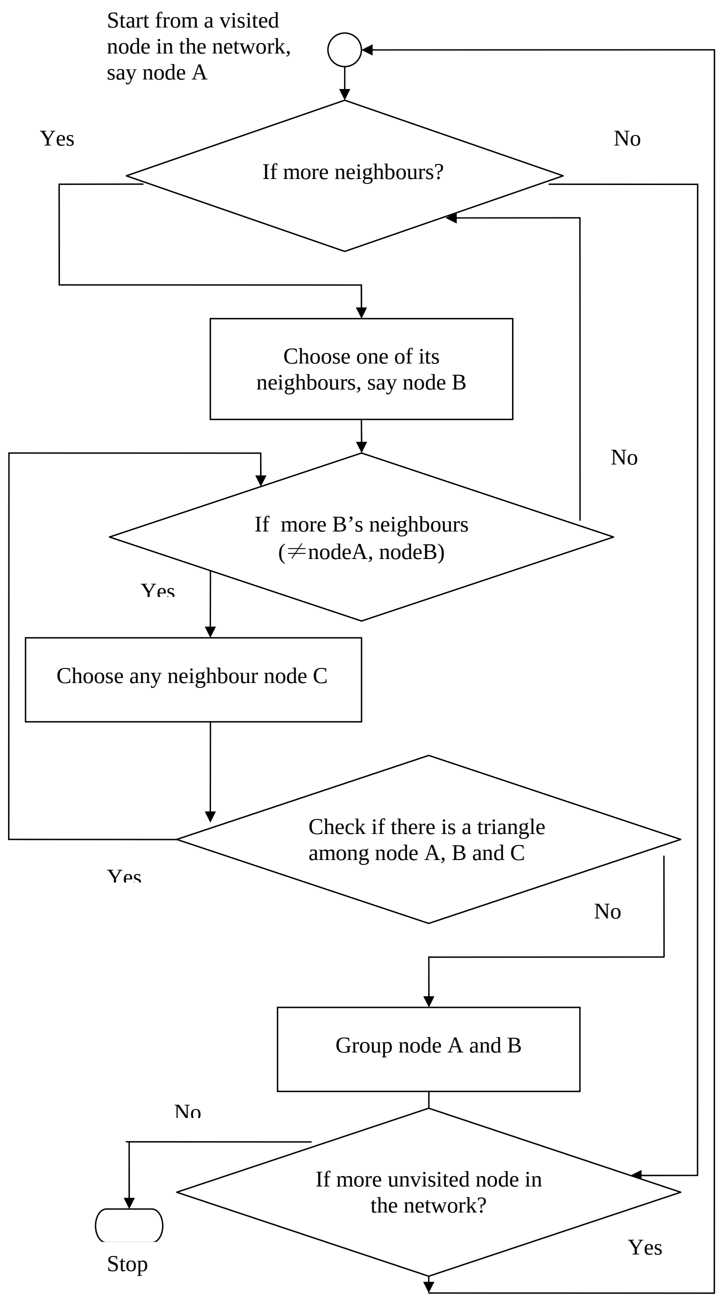


Figure 3.3: The flow chart for the procedure of the *deforestation* algorithm.

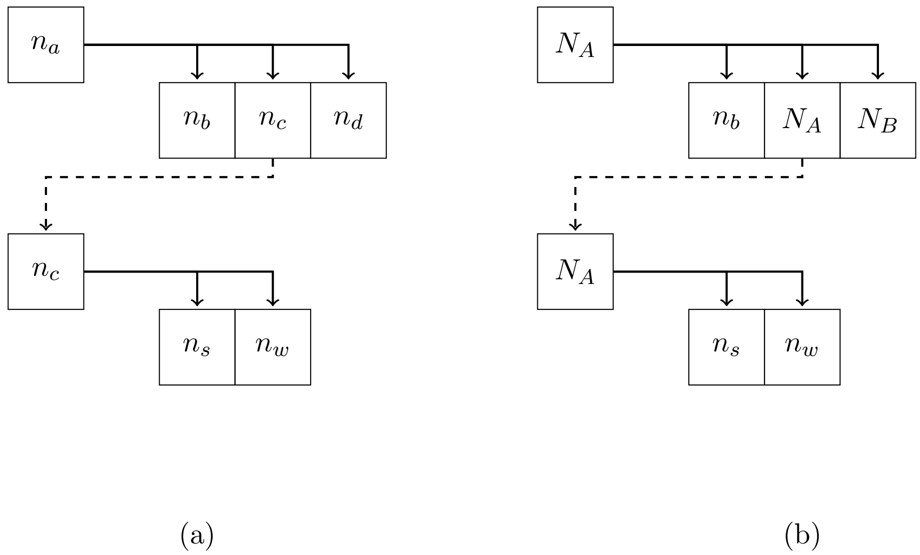


Figure 3.4: The network is represented as a linked list of nodes. (a) Node n_a has a pointer to a list containing its neighbours, n_b, n_c and n_d. The two nodes n_c connected by the dot line are actually the same nodes, which means node n_c has another two neighbours n_s and n_w, besides node n_a. (b) A super-node (N_A) is represented with a similar linked list but the nodes belonging to the super-node have the same name. For example, comparing with figure 3.4(a), node n_a and n_c are grouped into the super-node N_A in figure 3.4(b), so that node n_s and n_w are also the neighbours of the super-node N_A. The node n_d is grouped into the super-node N_B, so that it is represented by the name N_B.

Algorithm 1 Triangle(m, d)

Require: the mother node m, the daughter node d and a global list of daughters L
  $pointer \leftarrow p_m$ {check the daughters}
  if pointer is equal to pointer original daughter **then**
    $pointer_{nd} \leftarrow p_m + 1$ {pointer to new daughter n_{nd} }
  end if
  while there is a daughter in L **do**
    if the new daughter n_d and the original daughter d share a link **then**
      RETURN TRUE {they form a triangle}
    end if
  end while
  return FALSE {there is no triangle}

the neighbour are added to a LIFO stack to verify if there is a triangular relationship. The number of comparisons to check if there is a triangular relationship depends on the number of the first and second neighbours of node a, which is related to the degree of nodes. The algorithm is efficient for sparse networks. The worst case is when the network is a tree, in this case, each node should verify the triangle with its neighbours, which takes $2 * l$ steps.

Algorithm 2 Contraction(n, p)

Require: mother node n, pointer to daughters p and a global list L
  $L \leftarrow n_m$ {append mother node to temporary list}
  $pointer \leftarrow p_d$ {pointer to daughter node}
  if any more daughters **then**
    while there is the daughter in L **do**
      $pointer \leftarrow p_d + 1${pointer to next daughter}
    end while
    if there is a Triangle(mother, daughter, grandaughter) **then**
      while there is the daughter in L **do**
        $pointer \leftarrow p_d + 1$
      end while
    else
      $L \leftarrow n_d$ {append daughter to the list}
      Contraction(n_d, p_d)
    end if
  else
    return {no more daughters or a triangle}
  end if

Algorithm 3 Deforestation(G)

Require: the network G
  Choose a starting node
  while There is an unvisited node **do**
    $node \leftarrow n$ {the unvisited node.}
    $pointer \leftarrow p$ {get the position of the unvisited node in the linked list of nodes}
    Contraction(n, p)
  end while
  return FALSE {All the nodes have been visited.}

Estimate the size of skeleton networks. To obtain an estimate of the number of nodes and links in the skeleton network, prior to the contraction, we assume that the skeleton network consists only of triangles, that the network can be embedded in a torus which is tessellated with these triangles. With these assumptions the number of links will be three times the number of nodes, that is $L \leq 3N$. From the definition of the cyclomatic number $M(G)$, we know $M(G) = L - N + 1$ (see chapter 2). Therefore, the number of nodes N in the skeleton network is bound by

$$\begin{aligned} M(G) &= L - N + 1 \\ &\leq 3N - N + 1 \\ &\leq 2N + 1 \end{aligned}$$

$$N \geq (M(G) - 1)/2. \tag{3.1}$$

3.4.3 An example

Figure 3.5 shows an example of the *deforestation* algorithm. The original network is shown at the top. The nodes belonging to the super-nodes are represented in different colours in the original network (middle), and the skeleton network (bottom). Each colour (except white) represents a different super-node, while the white nodes are retained from the original network. The skeleton network has the same cyclic structure as the original network. In the figure the size of a super-node is proportional to the number of nodes contained inside the super-node. The more nodes contracted, the bigger the super-node is.

We also tested the *deforestation* algorithm using the AS-Internet graph and IPA-Internet graph. The results will be presented in chapter 4.

3.5 Basic features of the ***skeleton network***

From the previous example, we conclude that in the skeleton network the number of nodes and links of the skeleton network are reduced. The nodes inside a super-node form a tree, i.e. there are no alternative routes (no routing decision needs to be made) in the nodes contained in a super-node. The nodes of the skeleton network are a mixture of super-nodes and original nodes.

The skeleton network consists of only cycles, the majority short cycles, i.e. triangles. The

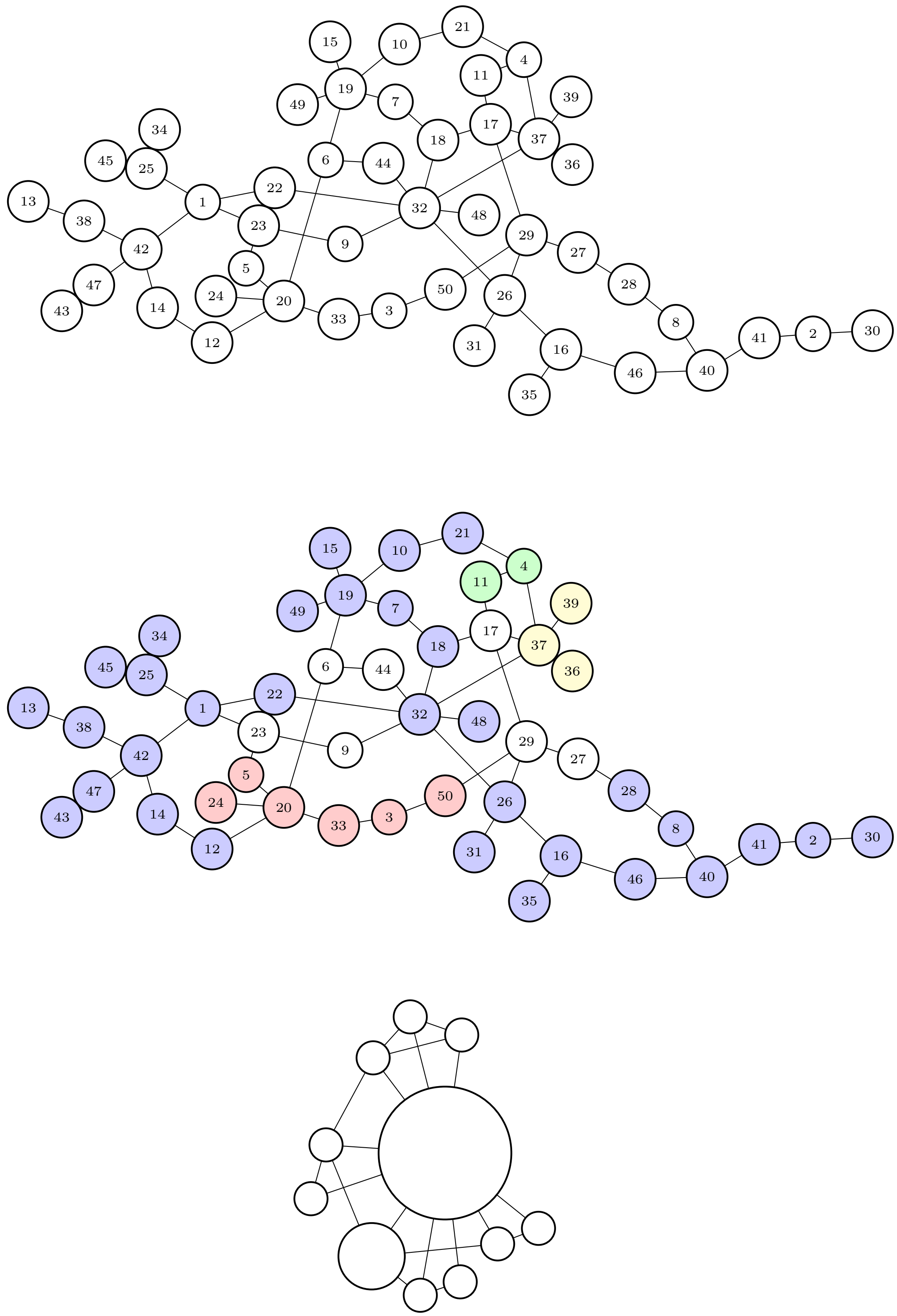


Figure 3.5: The *deforestation* algorithm simplifies a network, which originally has 50 nodes and 59 links to a network of 11 nodes and 20 links. The first graph at the top is the original network topology. The one in the middle shows how the *deforestation* groups the nodes into super-nodes. Each colour except white represents a different super-node, i.e. purple, green, pink, yellow. The white nodes are nodes retained from the original network. The graph at the bottom is *skeleton* network, which are the simplified representations of original network. It is reduced to 11 nodes and 20 links.

triangle is the simplest diversity unit. Therefore the skeleton network keeps the same path diversity as the original network. From the figure, it is easily to notice that the diameter [1] of skeleton network is smaller than the original network. That is, the network diameter is reduced.

Super–nodes can be considered as communities defined by the condition that there is only one path between any two members of the community. Hence the nodes inside the super–nodes can be visualised as hierarchies if depicted in a dendrogram.

In terms of degree of network size, the *deforestation* algorithm works more efficiently in sparse networks or networks with low clustering coefficient.

3.5.1 Uniqueness of the Topological Representation

One interesting question is if the connectivity of the skeleton networks is unique, that is, if two skeleton networks obtained by contraction using different initial nodes have the same connectivity. Two graphs that have the same number of nodes and are connected in the same way are called isomorphic [62, 65], see appendix A for more details. There are several ways to find out if two graphs are isomorphic, for example a necessary condition will be that the two graphs have the same number of nodes and the same degree distribution. However, in our case, it is very simple to show that the skeleton networks obtained from different initial nodes are not isomorphic. Figure 3.6 shows that the connectivity of the skeleton networks is not unique when the *deforestation* started from different initial nodes. The property that the skeleton network is not unique can work to our advantage so that we may construct a skeleton network that satisfies given network traffic constraints, and we explore this using extended *deforestation* algorithm that will be introduced in chapter 5.

[1] The ***diameter*** of a network is the longest shortest-path in this network.

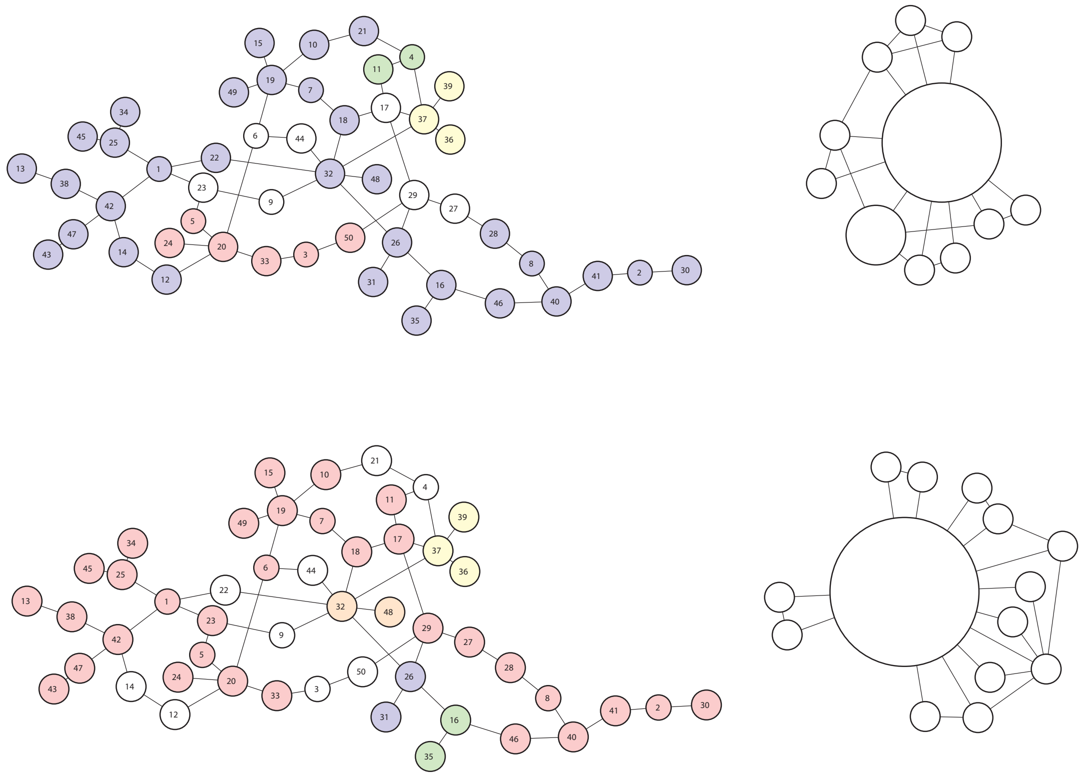


Figure 3.6: Deforestation of a network using different starting points. The skeleton networks (right) are not isomorphic.

3.6 Visualization

There exist some algorithms for the visualisation of large networks of 10000 nodes [35], however for very large networks, the visualisation is still an intractable problem. The *deforestation* algorithm could also be considered as a way of reducing the complexity of the graph layout problem, which can be done by grouping the nodes belonging to the super–nodes together when drawing the graph. Figure 3.7 shows the network of Figure 3.5 where the nodes belonging to the super-nodes are plotted near each other. The layout of the graph can be done using a force-direct method [33, 74]. The basic idea of the method is to represent the network as a graph of electric charges and springs. The charges (nodes)

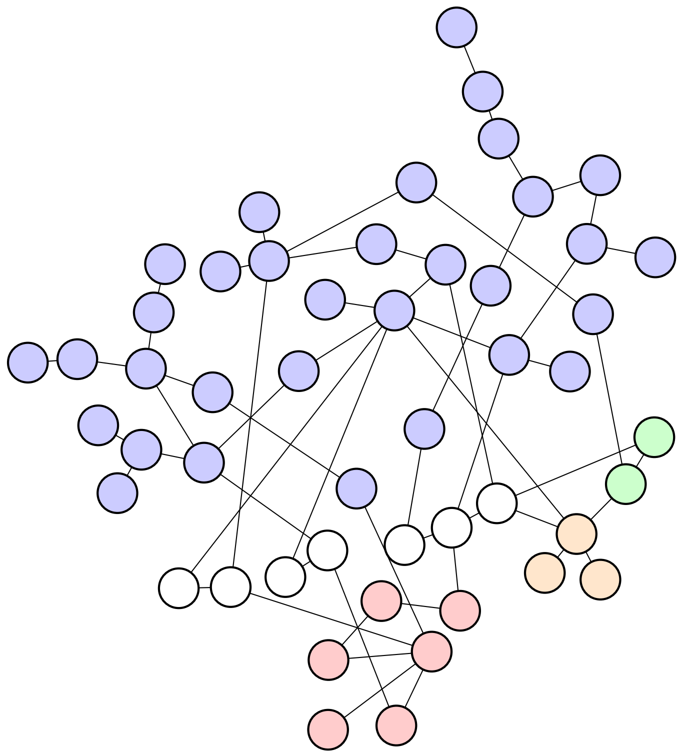


Figure 3.7: Graph of the network of figure 3.4 where the nodes belonging to the same super-node (labelled by color) are drawn near one another. This graph clearly shows how the nodes belonging to different super-nodes interact with each other.

repel each other while the springs (links) attract the charges. For each node, attractive and repulsive forces are evaluated with respect to the other nodes in the network. The positions of the nodes are changed according to the forces acting upon them until the network settles into a minimum-energy state. The graph layout is given by this minimum-energy state. In this procedure, strongly connected nodes appear close to each other, while weakly connected nodes are far apart. In this method the nodes are represented as repelling charges and the links as springs. The method finds the position of the nodes in space which minimises the total energy of the system. The method produces visually pleasing graphs.

3.7 Relationship with previous work

Large-scale networks containing hundreds of thousands of nodes are difficult to study. For example, in the visualisation of large networks, displaying all nodes and links provides no real benefits to understanding the network's connectivity. The density and complexity of the network overwhelms standard computer displays due to their limited resolution. Network simplification provides a way of making large network comprehensible, without removing the relevant structure of the network, so that modelling and research studies can be conducted on such network simplification, and then approaches and results could be extrapolated to Internet size networks. With the aim of making large-scale network simulation feasible and also reducing their complexity for visualization purpose, several methods have been proposed.

3.7.1 Simplification using Spanning-Trees

An algorithm widely used in telecommunications networks is the *Minimum Spanning Trees* (MST), which generates a simple tree to span (reach out to) all the nodes of the network. The MST algorithm can be considered as a way to simplify the network as all the other possible paths not contained in the spanning tree are ignored. The MST algorithm reduce the number of links used by the network, however the total number of nodes does not change at all.

Our simplification algorithm is also looking at "trees" of the network but in a reverse way. Instead of ignoring possible shortest-paths between each pair of nodes, all the alternative routes are kept in the network simplification. That's because in the traditional network routing, only single path is allowed to deliver information and alternative paths are considered to decrease the probability of blocking. However nowadays Internet applications and online services require high-bandwidth and best QoS (Quality of Service) for end-to-end delivery. And hence multipath routing is increasing in importance, i.e. for example for video streaming. Multipath routing exploits the physical network resources by utilising multiple alternative paths between a source-destination pair. This can yield a variety of benefits such as fault tolerance, increasing bandwidth or improve security.

3.7.2 Visual simplification by the k-core decomposition algorithm

Some related work has been proposed primarily on visualising the structure of complex networks. To the best of our knowledge, the algorithms presented by Baur *et. al.* [8] and Alvarez-Hamelin *et. al.* [4] are the only methods that are directly targeted to the study of large communications networks, i.e. the AS-Internet map. The algorithms presented by these authors are based on the *k-core* decomposition algorithm [7, 71], which consists in identifying particular subsets of the graph, called *k-cores*. These cores are obtained by recursively removing all the nodes of degree smaller than k, until the degree of all remaining nodes is larger than or equal to k. This algorithm is used as a visualisation tool for very large sparse networks, and it is easy to discover that larger values of "coreness" clearly correspond to nodes with larger degree and more central position in the network's structure. Comparing to our methods, we can not only preserve the core of network during the simplification but also keep the connectivity of the network.

3.7.3 Simplification via community algorithms

Simplification may also be accomplished through the use of community decomposition algorithms. Girvan and Newman [29] defined one clustering-based algorithm to visualise the community structure of networks. The simplified network obtained by the new simplification algorithm introduced in this chapter, just has a right property for applying such clustering-based algorithm. So that we can make use of this method to analyse and understand the community structure of the simplified network and do further reduction on it. We will fully discuss this approach in chapter 8.

3.8 Conclusion

In terms of network size, the *deforestation* algorithm simplifies the network graph efficiently by clustering the trees into *super-nodes*. The resulting *skeleton* network conserves the same path diversity as the original network. However the *deforestation* algorithm does not give a unique simplification of the network. We could take advantage of this variability of the skeleton network to select a skeleton network which satisfies some traffic constraints (chapter 5). Moreover, the grouping of nodes into the super–nodes are quite beneficial for the network visualisation.

Chapter 4

Deforestation of Real Networks

In this chapter we provide specific examples in which the *deforestation* algorithm is applied to the Internet at two different granularity levels. More precisely, we consider the Internet at the AS (Autonomous System) level and IPA (Internet Protocol Address) level.

4.1 Data sets of real networks

As introduced in the chapter 2, the Internet can be described at the AS level, at the router level and at the IPA level. For management purposes the Internet is divided into subnetworks, which are considered as an entity called an Autonomous System or AS. At the AS level the Internet can be considered in an abstract space where the relevant property is the connectivity between ASs. At the router level the nodes and links of the network represent physical entities. The nodes represent the routers and switches, and the links represent the different physical connections between nodes. Measuring the connectivity at the router level is a difficult and unresolved problem. Instead, the strategies are to establish a correspondence between IP addresses, domain names, and ASs using the *whois* database, which provides the connectivity of the registered headquarter's Internet Protocol (IP) interfaces, which is at the IPA level.

Here we consider two data sets describing the connectivity of the Internet: the AS level and the IPA level, referred as the AS-Internet and the IPA-Internet. Both the AS network data and IPA network data that we used are obtained from CAIDA[1] [10]. The AS-Internet is composed of approximately $11,000$ nodes and $23,000$ links. For the IPA-Internet, it is named as the macroscopic Internet topology data kit (ITDK) with the reference number of "$itdk0304$" and it was collected between April 21st and May 8th, 2003. The IPA-Internet network consists of around $200,000$ nodes and $600,000$ links.

4.2 Simplifying the AS-Internet network

Firstly, we review some basic statistical description of the Internet which was discussed in chapter 2.

The Internet has small-world properties, that is the average shortest path length among nodes is very small, normally 12 to 14. It has a heavy-tailed degree distribution so that the node degree distribution follows a power law decay, in which the high-degree nodes are preferred for connection by the rest of nodes and links. It has a rich-club phenomenon, that is, the high-degree nodes, also called "rich nodes", are very well connected to one another, which form the core of the Internet.

4.2.1 The size of skeleton networks

Applying the *deforestation* algorithm to the AS-Internet, figure 4.1 shows the distribution of the number of nodes for the AS-skeleton network when the contraction is started at different nodes. The distribution is well approximated with a Gaussian distribution. Table $4-1$ lists three typical reduced size of skeleton networks for AS-Internet: the minimum size, the most occurrence and the maximum size of the skeleton network. The contraction produces skeleton networks in a narrow range of sizes, so that we can take the most frequent solution as a good reduction of the original AS-Internet network. In this case the skeleton network

[1] www.caida.org

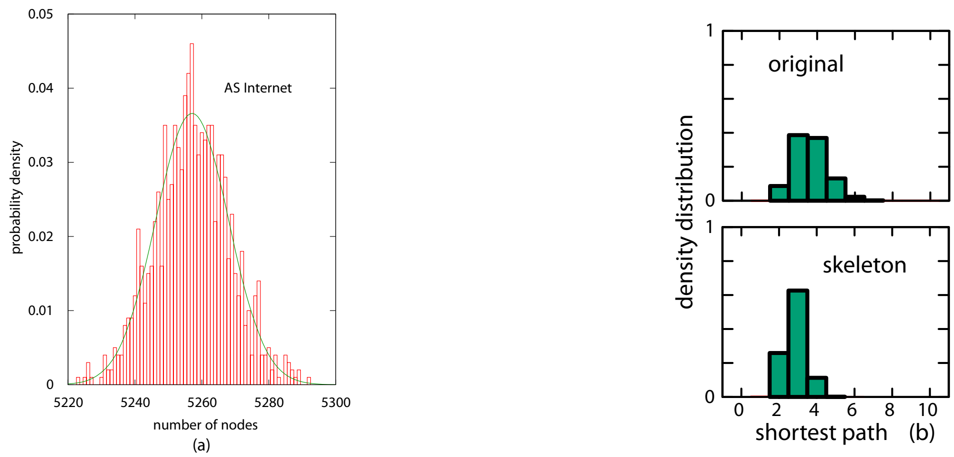


Figure 4.1: (a) Probability density of the number of nodes in the skeleton networks. (b) The length of shortest-path distribution for the AS-Internet and the AS-skeleton networks.

	Original	**Skeleton Network**		
		minimum	most occurrence	maximum
Nodes	11, 174	5, 223	5, 257	5, 292
Links	23, 409	17, 458	17, 492	17, 527

Table 4-1: Three typical reduced sizes of the skeleton network for AS-Internet are listed here: the minimum size, the most occurrence and the maximum size of the AS-skeleton network.

has approximately 50% fewer nodes and 36% fewer links than the original network. Also in figure 4.1, we show the skeleton network have the smaller diameter than the original network. In this case, the AS-network's diameter is reduced from 10 hops to 6 hops.

From Eq.(3.1) we obtain that the number of nodes of the skeleton network is around 6117 nodes. This result is slightly different from the network size obtained from the experiments, which is around 5250 nodes. In mathematics, the approximation is always a nondeterministic polynomial time (NP) hard problem, so that this result shows that Eq.(3.1) could give a reasonable approximation to the size of the skeleton network.

4.2.2 The correlations between the AS-Internet and AS-skeleton

Cyclomatic number. Besides figure 4.1(a), we also plot the probability density of the number of links in the skeleton network. The density of links has the same shape (Gaussian distribution) as the density of nodes. That because the statistics are conditioned by the cyclomatic number (introduced in Chapter 2):

$$\begin{aligned} M &= L - N + 1 \\ &= 23409 - 11174 + 1 \quad (original) \\ &= 17492 - 5257 + 1 \quad (skeleton) \\ &= 12236. \end{aligned}$$

The *deforestation* algorithm contracts the trees. The networks different number of alternative paths are related to the number of cycles. As the cyclomatic number of the original network and skeleton network are the same, the cyclical structure of the AS-skeleton network is the same as the original AS network.

Degree distribution. Figure 4.2(a) shows the degree distribution for the AS–Internet and its skeleton network both decay in similar fashion. This illustrates that the AS-skeleton network exhibits a statistical invariance with respect of the contracted trees. This is because for the AS-Internet, its skeleton network inherits the characteristics of its core. The AS nodes with highest degrees are well connected with each other, they tend to form the core of the network [83]. In the skeleton network, these highest degree nodes belong to different super–nodes. The links between these super–nodes are the same as the core of the AS network, so the skeleton network has the same core as the original network.

Statistics of super–nodes. The number of nodes inside the super–nodes also scale as a power law, see figure 4.2(b), that is there are many super–nodes that contain few nodes while there are some very large super–nodes. The reason for the existence of large super–nodes is because in the AS–network the nodes with high degree connect to many nodes of low degree, forming a star–like connectivity. Hence all these low-degree nodes are contracted into one super–node.

In the network we cannot assume that the importance of a node is simply related to its

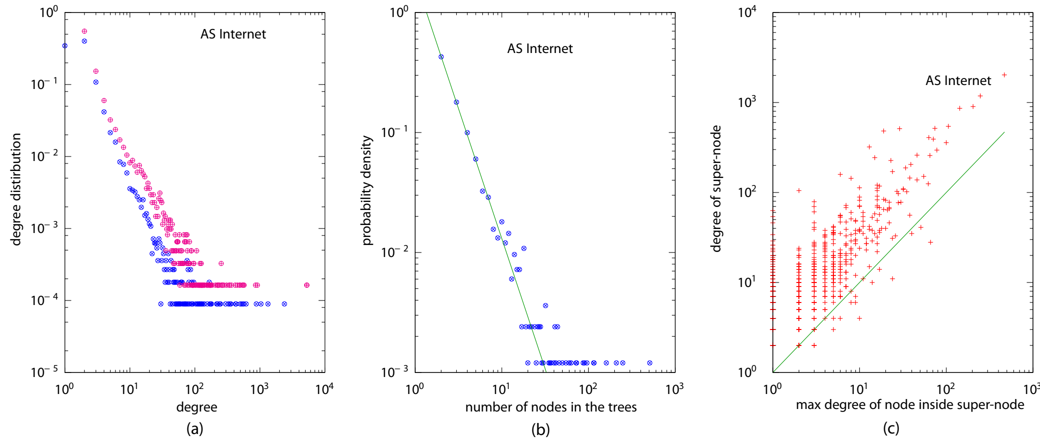


Figure 4.2: (a) Degree distribution for the AS-network (blue) and its skeleton network (red). The other graphs show the properties of the trees contained in the super–nodes. (b) Probability density of the number of nodes in a super–node. (c) Correlation between the maximum degree of a node inside the super–node and the super–node's degree. The green lines in figure (b) and (c) show the expected distributions which follow the power-law.

degree. The low-degree node may be an intermediate node, and has the same importance as the high-degree node that it connects to. Imagine that a low-degree node between two high-degree nodes acts as a *bridge* to transmit the traffic between them. If the low-degree node is broken down, the distance between the high-degree nodes should be considerably increased, or even worse that the network would be split into two parts. *Deforestation* algorithm collects high-degree node and its directly connected low-degree nodes into the same super-node, and may treat them as an integrated node because of the possible same importance for them in the network.

The distribution of the number of nodes in the super–nodes hints to a sort of global *self-similarity*[2] in the network structure of the Internet. This could have implications when modeling the dynamics of the network [79].

Figure 4.2(c) shows that the degree of the super–node and the node with maximal degree contained in the super–node are correlated, which implies that there is a strong correlation between the original network and the skeleton network. This correlation is due to the interconnectivity of the high degree nodes.

[2]In mathematics, a *self-similar* object is exactly or approximately similar to a part of itself.

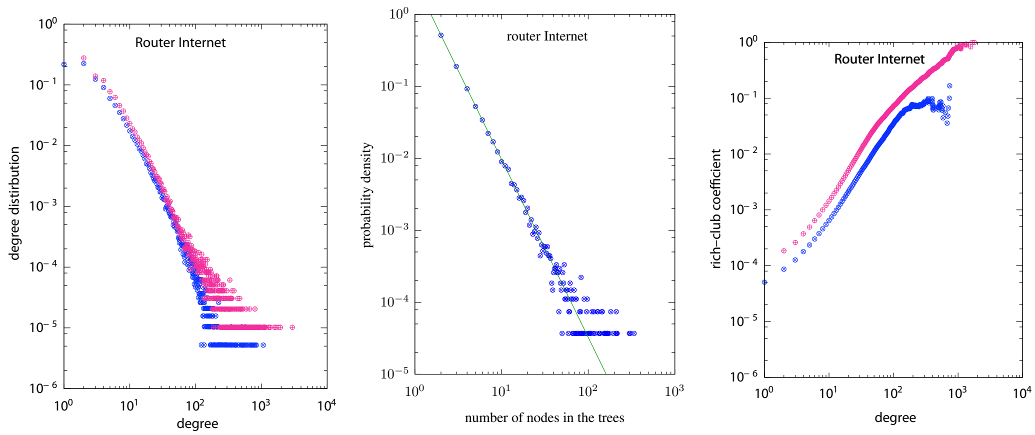
Triangles in the skeleton networks. In chapter 3, we expect that the skeleton network consists of cycles only, the majority are triangles. After the simplification, the average transitivity, here measured by the average link clustering coefficient, has increased from 0.89 for the AS-Internet to 1.29 for the AS-skeleton network. We counted that $11,516$ independent triangles in the skeleton network and $12,236$ cycles totally. The number of triangles comprises 94% percentage of number of cycles, thus the majority of cycles in the skeleton AS-Internet are triangles.

The statistical distribution of cycles has recently been acknowledged as particular by important for defining not only the topology of the networks, but also the dynamics of the system (i.e. network traffic) [5, 50]. In addition to that, Bagrow *et. al.* in [6] reveals that short cycle detection would help the identification of communities in complex networks. Hence the skeleton network that consists of only short cycles has many applications for the AS-Internet network, and we will discuss it in chapter 9.

The rich-club connectivity. The rich-club coefficient for AS-Internet and AS-skeleton network are similar. The rich-nodes are tightly connected, so that the *deforestation* algorithm cannot simplify them. These rich nodes are the core tier of the network structure, acting as super traffic hubs and providing a large selection of short-cuts for routing. Both networks (AS-Internet and AS-skeleton) have nodes with very large degrees, i.e. the maximum degree of the skeleton graph is $5,333$. Removing one of these nodes, the network would break into several unconnected components. For example, the maximum degree of the skeleton network is $5,333$. Removing this node and its links, the new cyclomatic number becomes $12,236-(5,333-1)=6,902$, which has a reduction of 56% in the number of cycles. In brief, it is very important for the AS-skeleton to conserve the rich-nodes of the AS-Internet, as in the AS-Internet the rich nodes are fundamental for the existence of alternative paths.

Original		Skeleton	
nodes	links	nodes	links
190,914	607,610	97,859	514,555

Table 4-2: The number of nodes and links for the IPA network, and its typical size of IPA-skeleton network.



(a) (b) (c)

Figure 4.3: (a) The degree distribution of the IPA–network (blue, lower) and its skeleton network (red, top). (b) Probability density of the number of nodes in a super–node. And the green line is the expected line which follows the power-law distribution.(c) Rich–club coefficient for the IPA–network (blue, lower) and its skeleton network (red, top), which shows rich nodes in the skeleton network are more tightly connected than the original network.

4.3 Simplifying the IPA-Internet network

We also applied the *deforestation* algorithm to the IPA-Internet.

4.3.1 The reduction of IPA-skeleton network

Table $4-2$ shows the typical reduction of the skeleton networks for the IPA-Internet. The IPA-skeleton network has approximately 50% less nodes than the original network. The decay of the degree distribution of the IPA–network and its skeleton behave like power-law, see figure 4.3(a). Similar as the AS-Skeleton, the number of nodes in the super-nodes also scale as a power-law (figure 4.3(b)), that is there are many super-nodes that contain few nodes, and there are some very large super-nodes.

4.3.2 The rich-club coefficient

Compared to the AS-Internet the high degree nodes in the IPA–network share very few connections. In chapter 2, we introduced the density of connections between the nodes with degree greater than k, is measured by the rich–club coefficient [83] $\phi(k) = 2E_{\geq k}/(N_{\geq k}(N_{\geq k} - 1))$ where $E_{\geq k}$ is the number of links between the nodes with degree higher than k and $N_{\geq k}$ is the number of nodes with degree higher than k. Figure 4.3(c) shows the density of connections for the IPA-Internet and its skeleton network. For the IPA-Internet the high degree nodes are sparsely connected in contrast with its skeleton network which are tightly connected.

This suggest that the rich nodes in the IPA Internet do not form a single core. We confirm this suggestion in chapter 8 when we evaluate the community structure of the IPA network.

4.4 The measurement of algorithm

We run our algorithms in a 16 core computer cluster with 48GB of RAM and 2GHz Woodcrest processors, and it took the *deforestation* algorithm less than one minute to simplify the AS graph, and approximately 10 minutes to simplify the IPA level network graph.

4.5 Conclusion

For the AS-Internet and IPA-Internet the *deforestation* algorithm generates a skeleton network which has approximately half the number of nodes of the original network. The cyclomatic number is retained, and hence the number of alternative path are conserved. The majority of cycles are triangles, up to 94% for AS-Internet. The distribution of nodes inside the super-nodes, the correlation between the degree of a super-node and the maximum degree of node contracted in that super-node scale as a power-law. This shows a strong correlation between the original network and the skeleton network. The rich-club connectivity in the AS-Internet exhibits very high for both original network and skeleton network, which means the very rich nodes in the AS-Internet are very tightly connected and they cannot simplify more during the reduction. These rich nodes may be a set of high hierarchical node and exist like a single core to act as a huge hub in the ”middle” of the Internet. However unlike the AS-Internet, the rich-club connectivity for IPA network changes after the reduction. The IPA-skeleton is more compact than its original IPA network. From the properties of the IPA-skeleton network it seems that in this case the network does not have a single core. The comparison between the statistical properties of the original network and its skeleton can reveal important features, like the existence of a core.

Chapter 5

Simplification with Traffic Approximation

As the connectivity of the skeleton network is not unique, this allows us to include other properties of the network when doing the contraction, in particular properties like capacity of the nodes and links. With this purpose, we are going to impose the conditions that we visit the network by considering the "importance" of links, where this "importance" is related to the usage of the links. An approximate measure of this "importance" can be obtained by *link betweenness centrality*, which has been introduced in Chapter 2 [26].

5.1 Estimation of traffic properties using betweenness centrality

Before showing the extended *deforestation* algorithm with link betweenness centrality, we would like to introduce how the network connectivity influences the dynamics of traffic flow and how to measure the traffic using network properties, i.e. betweenness centrality. A theoretical analysis for estimating the critical point for traffic transmission in the communications networks is presented as follow.

In a packet network, i.e. the Internet, the traffic is queued on all the nodes that it visits except for the destination node. To consider this situation we modify the betweenness centrality to

$$B(v) = \sum_{s \in N} \sum_{d \neq s \in N, d \neq v} p_{s,d}(v). \quad (5.1)$$

If the packets on the network are distributed evenly through all the shortest paths then the normalise betweenness

$$\hat{B}(v) = \frac{B(v)}{\sum_{i \in N} B(i)} \quad (5.2)$$

gives the proportion of usage of node v.

Link traffic

The *link betweenness centrality* $B(l)$ gives an estimation of the link load. If $B(l)$ can represent bits/second, packets/second or bandwidth and $B_{max}(l)$ is defined as the maximum load that link can carry, the $B(l)/B_{max}(l)$ is an approximation to the utilization of the link.

Node traffic

The *node betweenness centrality* $B(v)$ gives an estimate of the average traffic arriving at the node (traffic load). In this case, it is more useful to estimate the average size of the queues that the total amount of traffic the node can transmit (network load).

Mean Field Approximation to the network load

The critical load in terms of the betweenness centrality can be obtained using Little's law[39]. Little's law is a flow conservation law which can be stated that, in a steady state, the number of delivered packets (Pkt) is equal to the number of generated packets [27], or

$$\frac{d\,Pkt(t)}{d\,t} = \Lambda N - \frac{Pkt(t)}{\bar{\tau}(t)}, \quad (5.3)$$

where Λ is the average rate of packets generated per unit of time, $\bar{\tau}(t)$ is the average time that a packet spends in the system, and $Pkt(t)/\bar{\tau}(t)$ is the number of packets delivered per unit of time. Little's law does not depends on the arrival distribution of packets to the queue, the service time distribution of the queues, the number of queues in the system or upon the queuing discipline within the system. The law holds only when a steady state exist, that is below the critical load Λ_c. If the load is low and the delay $\bar{\tau}(t)$ is approximated using a constant delay $\bar{\tau}$, the queues on the nodes tend to be empty and the average delay time is the average shortest path $\bar{\ell}$, that is $\bar{\tau} \approx \bar{\ell}$. For higher loads the transit time can be approximated by the average shortest path plus the average time that a packet spends on the queues plus the service time (time in the system)

$$\bar{\tau} = \frac{1}{N} \sum_{i=1}^{N} T_i, \tag{5.4}$$

where T_i is the time spent in queue i, shortest-path plus the service time of the server tending that queue. If the network is not congested from the steady state solution $dPkt(t)/dt = 0$ gives

$$\bar{P}kt = \Lambda N \bar{\tau} = \Lambda \sum_{i=1}^{N} T_i. \tag{5.5}$$

To evaluate $\bar{P}kt$ in terms of the betweenness centrality we consider that the queues are all M/M/1 queues [18] with average arrivals λ_i, service rate μ_i and traffic intensity $\rho_i = \lambda_i/\mu_i$ then $T_i = 1/((1-\rho_i)\mu_i)$. Notice that the betweenness centrality and the average shortest path can be related by

$$\sum_{i=1}^{N} B(i) = N(N-1)\bar{\ell}, \tag{5.6}$$

Thus the average number of packets that arrive to node i is [82]

$$\lambda_i = \Lambda N \bar{\ell} \hat{B}(i) = \frac{\Lambda B(i)}{N-1} \tag{5.7}$$

where ΛN is the number of packets generated by unit of time by the whole network, $\bar{\ell}$ is the average shortest path of the network to account for the average number of packets that were produced in the past and they are still in transit. $\hat{B}(i)$ is the proportion of all the packets in transit that pass through the node i. The total number of packets on the network

$$\bar{P}kt = \sum_{i=1}^{N} \frac{\Lambda(N-1)}{\mu_i(N-1) - \Lambda B(i)}, \tag{5.8}$$

The Onset of Congestion

For high loads the majority of the packets of the network are on the busiest queue. If m labels the busiest queue then $\bar{P}kt \approx \bar{Q}_m$, at the congestion point $\bar{P}kt \to \infty$ and the critical load is

$$\Lambda_c = \frac{\mu_m(N-1)}{B(m)}, \tag{5.9}$$

which is the same expression obtained by Zhao *et al.*[82]. From Eq.(5.9), we know that the bigger the *betweenness centrality* is, the smaller the onset of congestion load. That means a node or a link with high betweenness would easily cause traffic congestion itself.

5.2 Deforestation extension

The extension of *deforestation* algorithm to include the link betweenness centrality can be easily take into consideration. The new procedure is to weight links by evaluating the betweenness centrality before the contraction. The contraction of nodes can follow the order of increasing importance of links (or decreasing importance of links). Thus the extended *deforestation* algorithm has two parameters, the initial node to start the contraction and a condition based on the flow. This condition will determine how the nodes are visited, hence how the contraction is done. First we contract the nodes which connecting links have a small flow. After no more contractions can be made, the flow-bound is increased, and the network is contracted again, and so on. In this way, we simplify the networks considering not only the network connectivity (topology) but also the approximate to the dynamics of the network (traffic).

The pseudo codes for the extended *deforestation* algorithm are shown below:

Algorithm 4 Deforestation(G)

Require: the network $G = (N, L)$

BetweenCentrality(void)
find the minimum and maximum value in centrality[], i.e. B_{max}, B_{min}
$gap = (B_{max} - B_{min})/N$
set $bound = B_{min} + gap$
Choose a starting node
while There is an unvisited node or the flow-bound beyond the maximum value of link betweenness **do**
  $node \leftarrow n$ {the unvisited node.}
  $pointer \leftarrow p$ {get the position of the unvisited node in the linked list of nodes}
  Contraction(n, p)
  $bound+ = gap$
end while
return FALSE {All the nodes have been visited.}

Algorithm 5 BetweenCentrality(void)

$centrality[] \leftarrow 0.0$
$linkCentrality[] \leftarrow 0.0$
$floyd(numberNodes)$ {calculate all the shortest path using floyd algorithm}
for $a \in 1, \cdots, source$ **do**
  for $b \in 1, \cdots, destination$ **do**
    searchTreeRoute$(maxlength, s, s, d)$
    for $k \in N$ **do**
      $centrality[k]+ = routesUsed/numberRoutes$
    end for
    for $k \in L$ **do**
      $linkCentrality[k]+ = linksUsed/numberRoutes$
    end for
  end for
end for

The algorithm of *searchTreeRoute* describes the procedures of discovering the number of shortest paths if given a source s and a destination d, and keep the track of links on each shortest path, accounting for caculating the link betweenness centrality.

The extended algorithm is slow for large networks because of the calculation of link betweenness centrality. However recently Brandes [9] has introduced a very efficient algorithm to evaluate the betweenness centrality.

Algorithm 6 searchTreeRoute($maxlength, v, s, d$)

Require: source s, destination d, the search starting node v, the length of shortest path between $(s, d) + 1$ $maxlength$

$partialList[] \leftarrow s, d$ {stack of the route between (s, d)}
$routesUsed[v] \leftarrow 0$
$linksUsed \leftarrow 0$
$numberRoutes \leftarrow 0$
$pointer \leftarrow v + 1$ {pointer to the node next to the s}
$newV \leftarrow pointer$
while there are more nodes between the source s and destination d and $newV$ exists **do**
  if the node $newV$ is already in the $partialList$ **then**
    if the node $newV$ is equals to the destination d and length of the $partialList$ equals to $maxlength$ **then**
      $numberRoutes++$ {found one route}
      for $i \in partialList$ **do**
        if length of $partialList \neq maxlength$ **then**
          $routesUsed[partialList[i]]++$
        end if
      end for
      for j is larger than 3 and $j \in partialList$ **do**
        for $k \in numberLinks$ **do**
          compare each link of route in $partialList$ to the $linkPairs$ {$linkPairs$ records two nodes end of each link}
          $linksUsed[k] += 1$
        end for
      end for
    end if
  else
    length of $partialList$ is not equals to $maxlength$ {the node $newV$ is not in the $partialList$}
    $partialList \leftarrow newV$
    searchTreeRoute($maxlength, newV, s, d$)
  end if$pointer++$ $newV \leftarrow pointer$
end while
RETURN

5.3 An example: extended deforestation in a small network

We apply the extended *deforestation* to the network shown in chapter 3. Figure 5.1 shows an example of the extended *deforestation* algorithm where the link betweenness centrality has been used as an approximation of the flow passing through the links.

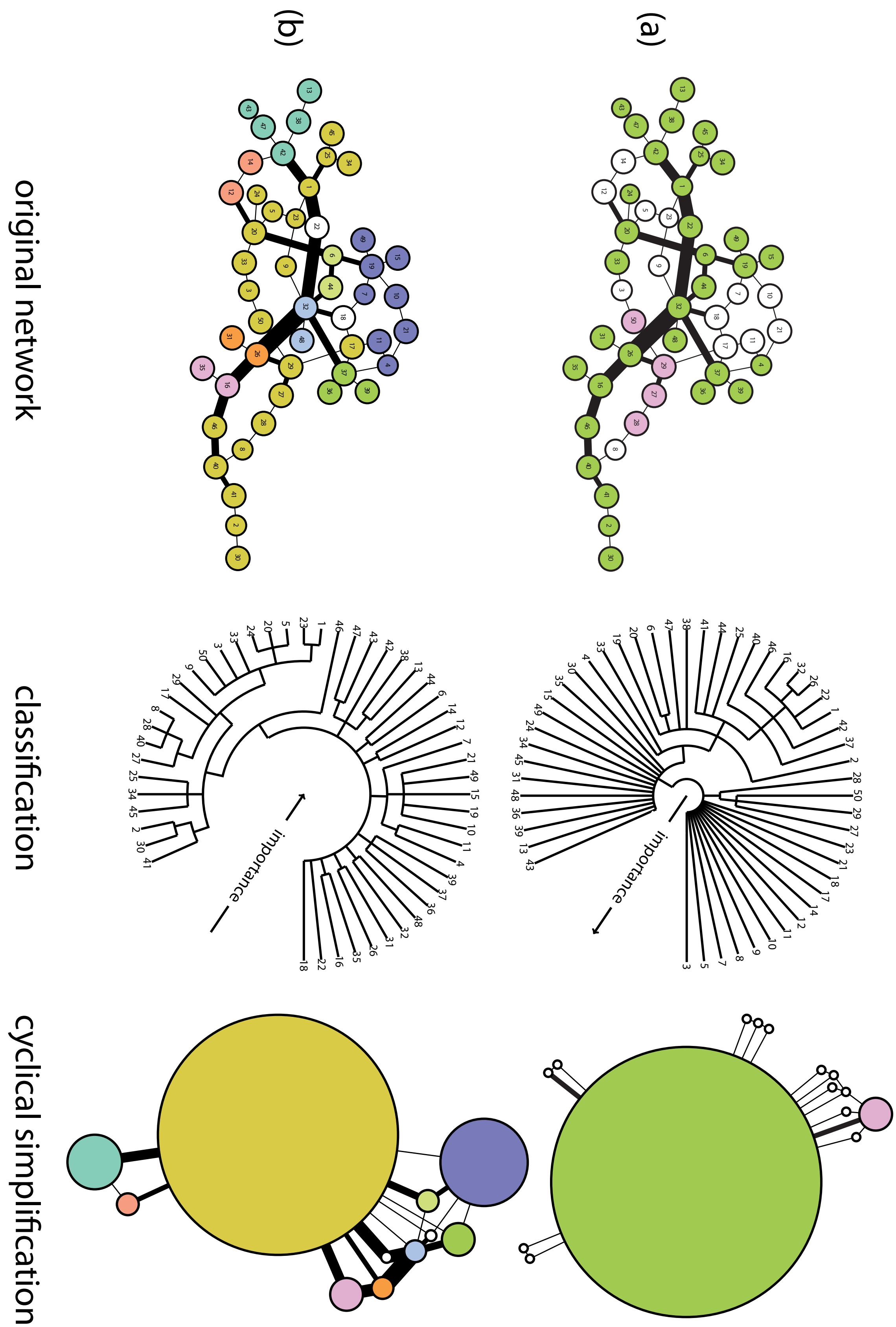


Figure 5.1: Extended *deforestation* contracts network following the "importance" of links. Nodes are contracted in the order of decreasing link-betweenness (a) and increasing link-betweenness (b).

5.3.1 The order of contraction

Figure 5.1(a) shows the original network, the order of contraction and its skeleton graph when the contraction is done by considering first the links with the largest betweenness centrality. The skeleton graph consists of a few nodes connected by “weak” links. All the important links are contained in the largest super-node. Figure 5.1(b) show the original graph, the order of contraction and skeleton network obtained by considering first the links with smallest betweenness centrality. In this case the skeleton network is a set of super-nodes connected by “strong” links. The skeleton networks obtained in this example represent the same network and have the same number of alternative paths, however they represent different view of the alternative paths. In figure 5.1(b) the majority of the links in the skeleton graph are important.

5.3.2 Communities & hierarchies

Directing the contraction by weighing the links (nodes) can be used to split the network into different hierarchies. The construction of the super-nodes will happen in different stages, each stage defining a hierarchy. The middle column of figure 5.1 shows the hierarchies of nodes in a super-node. In the dendrogram, a set of nodes is grouped together if the flow between their links is less than a given value. Moreover, nodes belonging to the same super-node can be considered as a community that are defined by having only one path between the members of the community.

5.3.3 The contraction of network for visualisation

The skeleton network displays cyclical structure of the network. The number of cycles implies of the number of alternative routes among the nodes, and they are retained in the skeleton graph. The size of a super-node is corresponding to the number of nodes collected in that super-node. Comparing the two contraction, figure 5.1(a) groups almost all the busy links and nodes into the “fat” super-node (shown in green) so that about 70% traffic may be transmited inside it. On the contrast, figure 5.1(b) distributes the busy links between the super-nodes.

5.4 The comparison between the ***deforestation*** algorithm and its extension

We applied the *deforestation* algorithm and its extension, i.e. the *deforestation* algorithm with link betweenness centrality, to a large network, and the skeleton networks are shown in figure 5.2. In the figure, the size of super-node is proportional to the number of nodes contained in the super-node, and the inside of the biggest super-node for each skeleton network are drawn at the bottom. If we assume that all the nodes in the both skeleton networks carry the same amount of traffic, the skeleton network obtained by *deforestation* algorithm with link betweenness centrality has a more evenly traffic-distributed in the super-nodes than the one by *deforestation*.

We also applied two algorithms to the Internet AS network. We noticed that the range of size of super-nodes is reduced when applying *deforestation* algorithm with link betweenness centrality, e.g. the biggest super-nodes obtained by *deforestation* contains 5333 nodes, while 3187 nodes for the extended algorithm. This shows that link betweenness centrality as a traffic approximation can guide the node contraction to produce a more balanced traffic-distributed skeleton network. The Internet IPA network has more than $600,000$ links and Brandes' algorithm [9] takes a very long time to calculate the betweenness centrality for all the links, so that we did not carry out this comparison in the IPA network.

5.5 Conclusion

In this chapter, we extended the *deforestation* algorithm to contract networks following the "importance" of links, which is considered as the approximation of traffic load for links. Therefore the extended *deforestation* algorithm has two parameters, the initial node to start

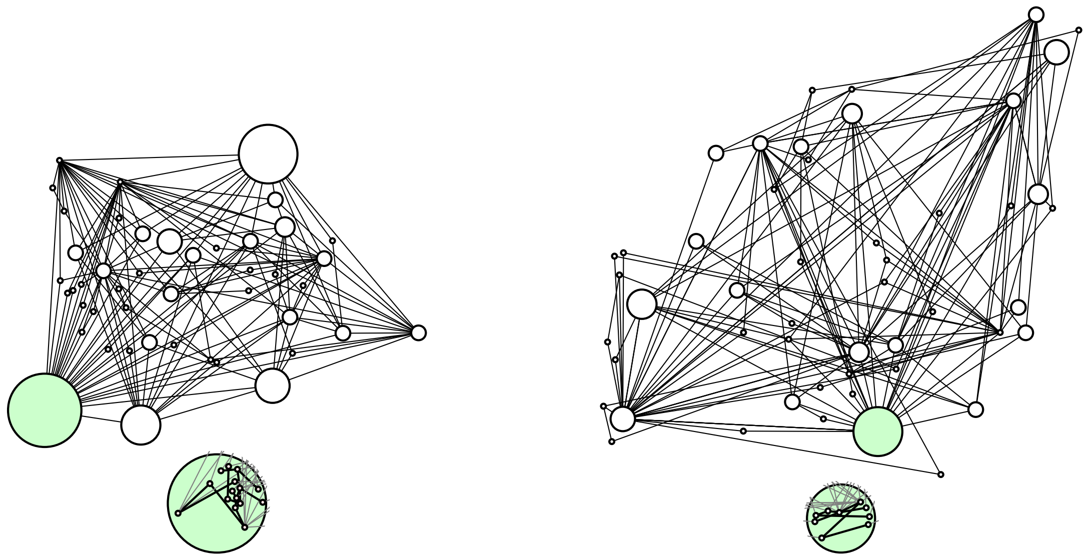


Figure 5.2: The comparison between the *deforestation* algorithm and its extension, i.e. the *deforestation* algorithm with link betweenness centrality. The skeleton network obtained by *deforestation* algorithm is shown on the left, while the skeleton network on the right is obtained by its extension. The biggest super-nodes (the green nodes) for each skeleton network are drawn at the bottom. And the size of the super-node is proportional to the number of nodes that it contains.

the contraction and a condition based on the flow. Different directions of contracting nodes would obtain a different skeleton graph. And the process of the network contraction will happen in several stages, so that it can be used to split the network into different hierarchies. Same as the *deforestation*, the extended *deforestation* also has made sizeable reduction, and provide a clear cyclical network structure for analysing and visualising. However it can produce a more balanced traffic-flow distribution in the super-nodes than the *deforestation*, and would be good for parallel computing when simulating a very large communications network.

Chapter 6

Traffic and Topology

6.1 Introduction

When studying a large-scale telecommunication network, a network-wide view of the traffic demands is needed. Shifts in user behaviour, changes in routing policies, and failures of network elements can result in significant fluctuations in traffic flows. This leaves network operators trying to tune the configuration of the network to adapt to changes in the traffic demands. In this chapter, we review how the topology and the traffic are related via routing protocol and estimate the network traffic via the traffic demand model. In the last chapter, we employ the betweenness centrality to approximate the traffic in the network and to guide the *deforestation* algorithm to simplify the network. Comparing with betweenness centrality, we would also like to extend the *deforestation* algorithm with the traffic demand estimate when doing the simplification. In this chapter we introduce the background and justification for this, and the example of medium-scale scenario is shown in the next chapter (Chapter 7).

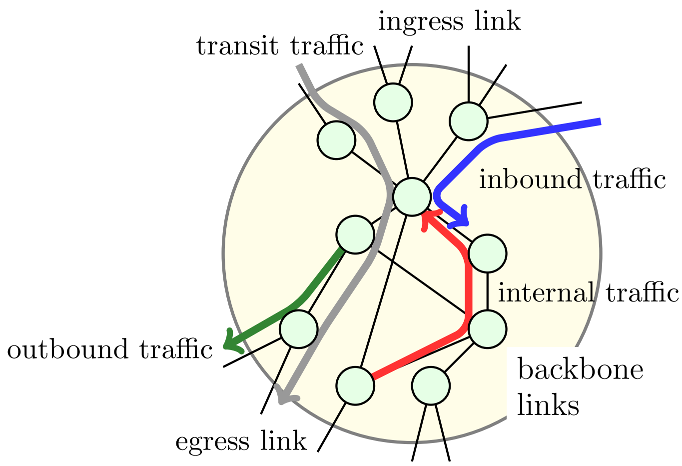


Figure 6.1: Four possible traffic scenarios

6.2 Traffic flow through ISP backbone networks

First, we have a brief overview of the Internet Service Provider (ISP) backbone architectures and routing protocols. A backbone network consists of a collection of routers and links. All the links are bi-directional, and they are usually divided into backbone links and edge links. Backbone links connect routers inside the ISP backbone, while edge links connect to downstream customers or neighbouring providers. When carrying traffic into the ISP backbone, an edge link is called an ingress link; when carrying traffic away from the ISP backbone, the link is called an egress link. Figure 6.1 illustrates the four possible scenarios:

- *internal traffic* that travels from an ingress link to an egress link within one domain or provider;
- *transit traffic* that travels from an ingress link in a neighbouring domain to an egress link in another neighbouring domain;
- *inbound traffic* that travels from an ingress link in the neighbouring domain to an egress link within the domain;
- *outbound traffic* that travels from an egress link within the domain to an egress link in the neighbouring domain.

Much of the traffic in the Internet must travel through multiple domains. The interplay between intra-domain and inter-domain routing has important implications for how we define traffic demand. The ISP employs an intra-domain routing protocol, such as Open Shortest-Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS), to select paths through the backbone. The routers exchange link-state information and forward packets along shortest paths, based on the sum of link weights chosen by the ISP. Communicating across domains requires the exchange of reachability information. The Border Gateway Protocol (BGP) is used to exchange dynamic reachability information with the remaining customers and neighbouring provider. The ISP backbone network lies in the "middle" of Internet, and may not have a direct connection to the sender or the receiver of any particular flow of packets. Also an ISP may have multiple links connecting to a neighbouring provider. When a router learns multiple routes to the same destination, the ultimate decision of which route to use depends on the BGP route-selection process, which considers the length of the path, in terms of the number of autonomous systems involved, followed by several other criteria [31, 73].

6.3 Traffic demand model

How should traffic demands be modelled and inferred from network measurements? At one extreme, the network traffic could be represented at the level of individual source-destination pairs, possibly aggregating sources and destinations to the network address or autonomous system level. Such an end-to-end traffic matrix is one of the existing measurements. Other techniques would provide views of the effects of the traffic demands, such as end to end performance (e.g. high delay and low throughput) and heavy load (e.g. high link utilization and long queues). These effects are captured by the measurements of delay, loss, or throughput on a path through the network. To be practical, the representation of traffic demands should enable experimentation with changes to the network topology and routing configuration.

6.3.1 Traffic demand estimation

Traffic matrix reflects the volume of traffic demands that flows between all possible pairs of sources and destinations in a network. The knowledge represented by a traffic matrix is very valuable to a wide variety of *traffic engineering* tasks including load balancing, routing protocols configuration, dimensioning, provisioning and fail-over strategies [30, 45].

In the networking scenario, we can assume that a variety of application can be delivered across the network, like audio (VoIP), video conferencing and FTP file transfers.

Connections are set up between the traffic sources and the ingress edge routers. New connections are established at each edge router with exponentially distributed inter-arrival times $\triangle t$ with a mean $\overline{\triangle t}$. That means, the creations of connections at each edge router follow *Poisson* process, in which, the inter-arrival times $\triangle t_1, \triangle t_2, ..., \triangle t_n$ are independent of each other and each have an exponential distribution with mean $1/\lambda$, where λ represents the average number of connections per unit time. We call λ, the connection established intensity, and we have

$$\lambda = 1/\overline{\triangle t}. \tag{6.1}$$

Holding time H of audio and video connections are also exponentially distributed with a mean $\overline{H}$, usually 50 seconds per connection for the average holding time of the audio, 100 seconds per connection for the video, and the holding time of FTP transfer sessions is variable by the file size Φ and mean sending rate.

Given the connection established intensity λ and average holding time $\overline{H}$, we can calculate the average number of simultaneous connections $\overline{N}$ mapped onto a network physical routing path for a certain source to destination pair (s, d) by using *Little's Law*, which states that "the average number of customers in a stable system (over some time interval) is equal to their average arrival rate, multiplied by their average time in the system" [39]. Then we have

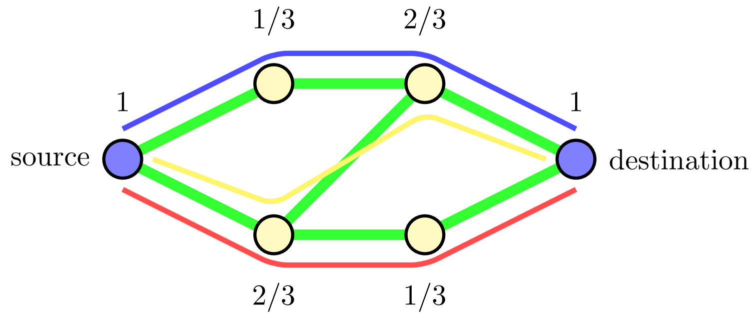


Figure 6.2: Traffic splitting for source s to destination d accordingly to ECMP rule.

$$\overline{N} = \lambda \cdot \overline{H}. \tag{6.2}$$

6.3.2 Traffic and routing

In the OSPF routing context, each node is aware of the weights $W(w_1, w_2, ..., w_n)$ of the links, which are evaluated by traffic load of individual links. The traffic is routed along the *shortest paths* which can be determined based on the link's weights $W(w_1, w_2, ..., w_n)$. Additionally, the *Equal-Cost Multi-Path* (ECMP) rule is used to distribute the traffic on several possible routes. ECMP states that given a graph $G(V, E)$, a source node s and destination node d. All the packets arrive at source node s are directed along the shortest path to destination d. If there is more than one link outgoing from s and belonging to the shortest path from s to d, then the packets will be distributed evenly among these links.

Figure 6.2 shows a simple network, where has three shortest paths between source s and destination d. According to the ECMP rule, the traffic between nodes (s, d) will be equally split and distributed among these three paths. We denote this ratio of traffic splitting as the critical factor α. The critical factor of a physical network link is also considered as the betweenness centrality of that link (already defined in Chapter 5). The bigger α of a physical link implies more flows routed through that link.

Suppose the inter-arrival times and holding times of the connections follows an exponential distribution with mean of $\overline{\triangle t}_i$ and $\overline{H}_i$. According to *Little's Law*, we know that the average number of simultaneous connections $\overline{N}_i$, is

$$\overline{N}_i = \lambda_i \cdot \overline{H}_i. \tag{6.3}$$

We assume that there are γ sources of each type of traffic connected to the source edge router, so the traffic load Ψ for each traffic class on an arbitrary link l can be calculated as

$$\Psi_{li} = \alpha_i \cdot (\gamma_i \cdot \overline{N}_i \cdot \bar{\mu}_i) = \alpha_i \cdot (\gamma_i \cdot \lambda_i \cdot \overline{H}_i \cdot \bar{\mu}_i), \tag{6.4}$$

where α is the critical factor percentage of traffic splitting by using different links, and $\bar{\mu}$ is the mean sending rate of each traffic source.

Thus, based on the observation above, we can derive the traffic demand matrix $\mathbb{M}$ of the overall network as

$$\mathbb{M} = \mathbb{M}_\alpha \cdot (\gamma \cdot \lambda \cdot \overline{H} \cdot \bar{\mu}), \tag{6.5}$$

where $\mathbb{M}_\alpha$ is the matrix of α_{ij}, as

$$\mathbb{M}_\alpha = \begin{bmatrix} 0 & \alpha_{1,2} & \alpha_{1,3} & \cdots & \alpha_{1,j} & \alpha_{1,j+1} & \cdots & \alpha_{1,N-1} & \alpha_{1,N} \\ \alpha_{2,1} & 0 & \alpha_{2,3} & \cdots & \alpha_{2,j} & \alpha_{2,j+1} & \cdots & \alpha_{2,N-1} & \alpha_{2,N} \\ \vdots & \vdots & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \alpha_{i,1} & \alpha_{i,2} & \vdots & 0 & \cdots & \alpha_{i,j} & \alpha_{i,j+1} & \cdots & \alpha_{i,N} \\ \alpha_{i+1,1} & \alpha_{i+1,2} & \vdots & \vdots & 0 & \cdots & \alpha_{i+1,j+1} & \cdots & \alpha_{i+1,N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \cdots & \cdots \\ \alpha_{N-1,1} & \alpha_{N-1,2} & \alpha_{N-1,3} & \vdots & \vdots & \alpha_{N-1,j} & \vdots & 0 & \alpha_{N-1,N} \\ \alpha_{N,1} & \alpha_{N,2} & \alpha_{N,3} & \vdots & \vdots & \alpha_{N,j} & \cdots & \alpha_{N-1,N} & 0 \end{bmatrix} \tag{6.6}$$

However, representing all hosts or network nodes would result in an extremely large traffic matrix.

6.4 Deforestation extensions

The deforestation algorithm contracts the network, taking into consideration only the connectivity of the nodes (see chapter 3) and considering both connectivity and traffic approximation of links (see chapter 5). This allows us to include other properties of the network when doing the contraction, in particular, like direction of the links and traffic demand matrix.

Extension with directions. The modification of the *deforestation* algorithm to include the links direction is straightforward. In the algorithm a link (b, c) means node b connects with c, but not vice versa.

Extension with traffic demand matrix. The inclusion of property like the traffic demand matrix can also be easily taken into consideration. The extension is very similar to the *deforestation* with traffic approximation in chapter 5. In this case the capacity (weight) of links are determined by traffic demand matrix. The **while** statements in the **Contraction** algorithm (algorithm 2 in chapter 3) are changed to include the link capacity, and a condition based on the capacity will direct how the contraction is done. For example, first we contract the nodes which connecting links have a small capacity, and then links with a higher capacity till no more contraction can be made.

6.5 Summary

By looking at the ISP backbone, there are four types of traffic scenarios. It gives us an idea to extend the *deforestation* with traffic direction. And in the traffic engineering, the traffic demand matrix is considered as an initial way to model the network traffic. It can also be included during the network contraction.

Chapter 7

Deforestation of the Medium-scale Scenario

7.1 Introduction

The purpose of this chapter is to investigate how *deforestation* algorithms are applied to a medium-scale network scenario. Given a medium-scale network, it can be represented in two ways: the logical scenario and the physical scenario. We are going to simplify both logical and physical scenarios using a traffic approximation (see chapter 5) and also using traffic directions and demands (see chapter 6). It is worth simplifying the network in different cases in order to fully illustrate the network contraction by *deforestation*.

7.2 Medium-scale network scenario

Figure 7.1 shows the scenario of the medium-scale network. This network is from [81]. At the end of this chapter, we would like to compare our results with the experiment results from [81], so that the properties of network are all kept as follows. This irregularly connected network consists of nine backbone nodes and fifteen bi-directional links. It is assumed

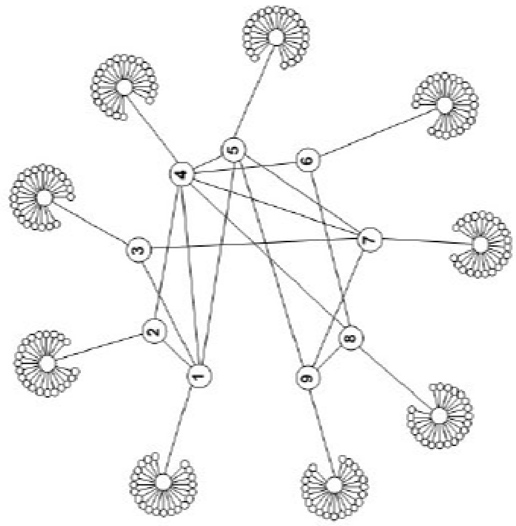


Figure 7.1: The medium-scale network scenario [81]

that each backbone node in the network is connected to one or more sub-networks, where traffic flows originate and terminate. The physical links connecting the backbone nodes (routers) within the network have capacity of 2048kps (E1 link) with fixed propagation delay of 10ms, and local high-speed links interconnecting the backbone nodes and sub-network nodes operate at 155Mps (OC-3) with fixed propagation delay of 1 microsecond. This ensures that network congestion occurs only within the network core. The network is configured as an OSPF (Open Shortest Path First) domain and we use the hop count metric as a benchmark for default shortest paths. Because of the speed difference between backbone links and local links, there is no traffic control included at the edge of the domain. Also here it is assumed that it's a single-service network, where it consists only of one type of traffic.

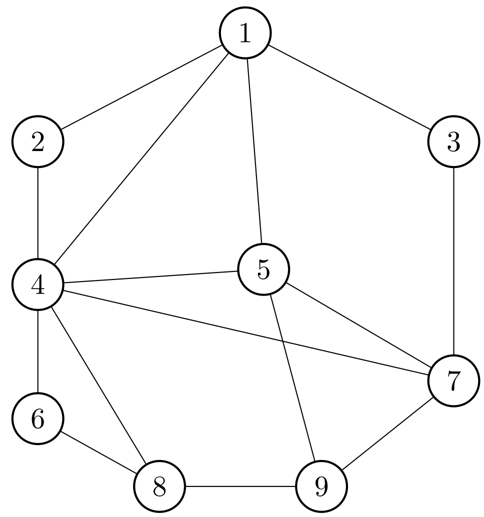


Figure 7.2: Logical representation of medium-scale network

7.3 Logical scenario

7.3.1 Representation of logical scenario

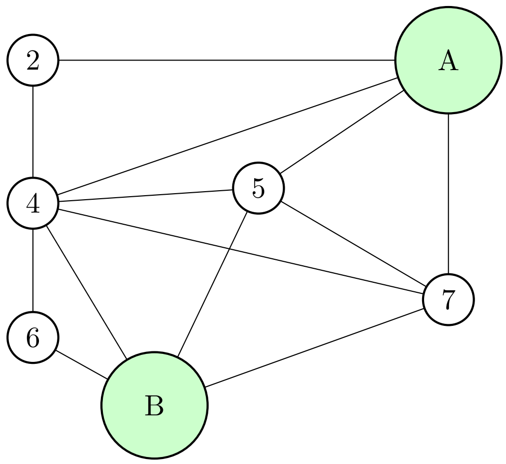
The medium-scale scenario in figure 7.1 has nine backbone nodes, fifteen bi-directional backbone links. The local high-speed link between each backbone node and its star sub-network could be thought as one node. Therefore, the nine backbone nodes are considered not only as router nodes, but also being a place where traffic flows originate and terminate. Thus the logical representation of the medium-network is shown in figure 7.2.

Applying the *deforestation* algorithms to the logical scenario of the medium-scale network are studied as follows.

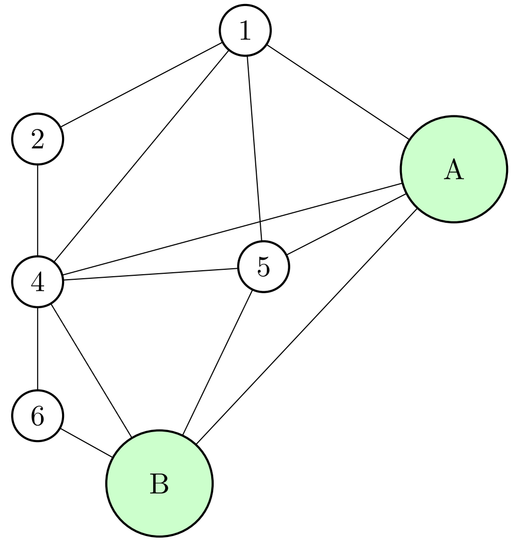
7.3.2 Results for logical scenario

Case 1 **- Deforestation algorithm**

In chapter 3, we discussed how the *deforestation* algorithm could produce skeleton networks with different network connectivities if the starting nodes are different. For the logical scenario of a medium-scale network, there are two different network simplifications, see figure 7.3. Both skeleton networks are simplified to a network of 7 nodes with 13 links.



(a)



(b)

Figure 7.3: Two simplifications of logical scenario

However the network nodes are grouped differently when looking inside the super-nodes A & B. The figure 7.3(a) groups node 1, 3 and node 8, 9 into super-nodes, while in figure 7.3(b), super-node B is contracted in the same way but super-node A aggregates the node 3, 7 instead.

Back to the original network shown in figure 7.2, there exist 2 quadrangles, i.e. $\{1, 3, 5, 7\}$ and $\{4, 5, 8, 9\}$, which have the potential to be simplified into a triangle. Considering their positions with other nodes, the quadrangle $\{4, 5, 8, 9\}$ has only one way to contract whereas there are two ways of contraction for quadrangle $\{1, 3, 5, 7\}$, which are the same as the skeleton networks shown in figure 7.3.

Case 2 - Deforestation algorithm with link betweenness centrality

In this case, the *deforestation* algorithm is extended to impose the link *betweenness centrality* as a condition of network contraction.

Before the *deforestation* is performed, the *betweenness centrality* (B) for the network nodes and links are calculated, see table $7-1$. In this case, the *deforestation* only returns one

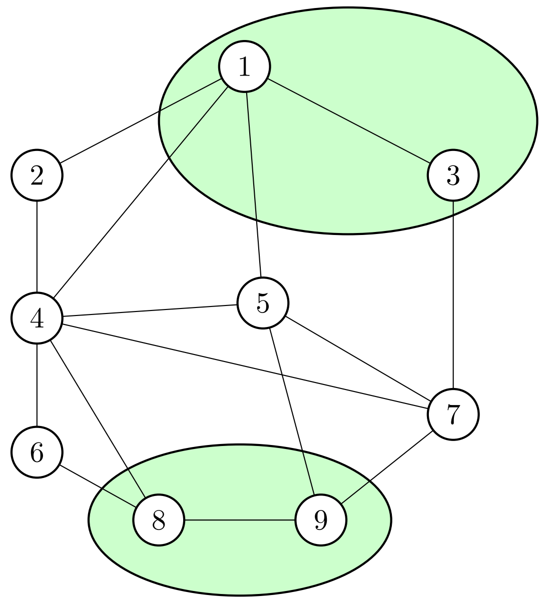


Figure 7.4: The skeleton network of logical scenario obtained by the *deforestation* algorithm with betweenness centrality

network simplification no matter which starting node is chosen, shown in the figure 7.4. In the graph, the node $8, 9$ and node $1, 3$ are grouped separately into the super-nodes.

The reason that there is only one simplification obtained by the *deforestation* algorithm with link betweenness centrality is easy to understand. From the extended algorithm itself, the links with low *betweenness centrality* are always contracted before the links with a high value.

In figure 7.4, the quadrangle $\{1, 3, 5, 7\}$ has two ways of contraction into the triangle, either contracting the link $1-3$ or contracting the link $3-7$. From the table $7-1$, the betweenness centrality for the link $1-3$ is 8.33, while the value for the link $3-7$ is 9.00. According to the algorithm, the link with low betweenness is contracted before the link with a higher value, then the only one network simplification is obtained when the contraction is done.

Therefore the link betweenness centrality, which approximates the utilisation of the links, can be considered as one of the factors that could guide the network contraction.

Case 3 - The deforestation algorithm with traffic demand matrix

Nodes	B
1	15.17
2	8.00
3	8.67
4	30.50
5	12.33
6	8.00
7	15.50
8	11.17
9	10.17

Links	B
Nodes - Nodes	
$1-2$	5.50
$1-3$	8.33
$1-5$	7.17
$2-4$	10.50
$3-7$	9.00
$4-5$	7.17
$4-6$	12.00
$4-7$	11.50
$4-8$	10.50
$5-7$	3.67
$5-9$	6.67
$6-8$	4.00
$7-9$	6.83
$8-9$	7.83

Table 7-1: The betweenness centrality for network nodes and links

When real traffic is applied to a network, it does not follow a uniformed traffic distribution, and these fluctuating traffic flows can be defined using a traffic demand matrix. A total of 72 source-destination paths are established to carry the traffic between any pair of nodes in their 9-node example.

Here we are going to consider two traffic demand matrices, which are constructed in different ways to represent different levels of network traffic dynamics. For the first traffic demand matrix ($TDMatrix_1$), the amount of traffic at each source-destination pair follows the same traffic distribution ($Poisson$) except for the node 1. The traffic flows traversing node 1 are three times as the traffic flow for the rest of the nodes pairs. The second traffic demand ($TDMatrix_2$) is also a symmetric matrix, as the logical scenario that we used here is a undirected network. For a source-destination pair the amount of traffic is related to the smaller label of the pair, i.e. if the packets sent by source 1 is assumed to be a unit of the traffic, then a source-destination pair $(i, i+1)$ and $(i+1, i)$ would sent i times of that unit over the link between the pairs.

$$TDMatrix_1 = \begin{pmatrix} 0 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 3 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 3 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

$$TDMatrix_2 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 2 & 0 & 3 & 3 & 3 & 3 & 3 & 3 \\ 1 & 2 & 3 & 0 & 4 & 4 & 4 & 4 & 4 \\ 1 & 2 & 3 & 4 & 0 & 5 & 5 & 5 & 5 \\ 1 & 2 & 3 & 4 & 5 & 0 & 6 & 6 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 & 0 & 7 & 7 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 0 & 8 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 0 \end{pmatrix}$$

As expected, the two traffic demand matrices produce different skeleton networks. The skeleton network for the ($TDMatrix_1$) is show in figure 7.5, which groups node $3, 7$ and $8, 9$ into super–nodes. While for the ($TDMatrix_2$), the skeleton network is same as the network simplification shown in figure 7.4, which contracts node $1, 7$ instead of $3, 7$. That's because in the first traffic matrix, node 1 generates three times more packets than the other nodes. The links connecting to node 1 are busier than the rest of links, and they are to be contracted later than other links. When the traffic demand matrices apply to the logical scenario, it estimates the fluctuating traffic as it travels the network, and shows that the traffic dynamic can change the order of the network contraction.

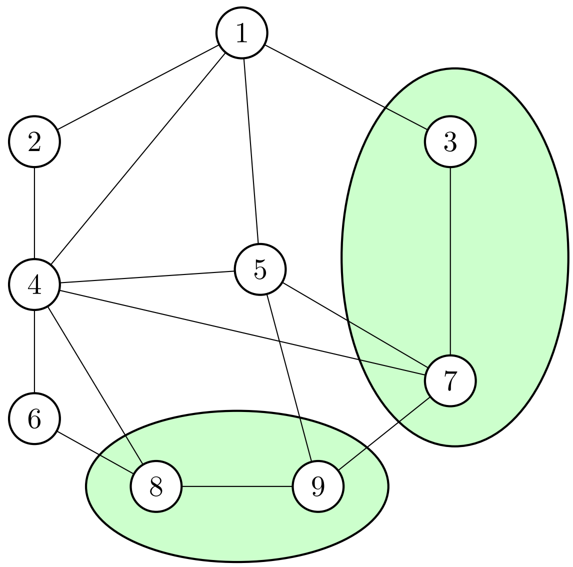


Figure 7.5: The skeleton network of the logical scenario is obtained by the *deforestation* algorithm with traffic demand matrix, when applying $TDMatrix_1$.

In summary, in real networks, the traffic demands may not be as simple as our scenarios, and also the measurements to obtain the traffic matrix could be very complicated. We used this simple example to explain how the *deforestation* algorithms are used to simplify the network, with the traffic approximation (link betweenness) and the estimate of traffic dynamic based on the demand matrix, and we notices that the traffic characteristics are fundamental for the simplification.

7.4 Physical scenario

The physical scenario reflects how the queues are working for each of the router nodes, how busy they are, and how traffic flows go through a certain link on different directions. The physical scenario of the medium-scale network (figure 7.2) is shown in figure 7.6.

7.4.1 Representation of physical scenario

According to [81], the backbone links are all output links, and the queues only exist on the direction that the backbone nodes (routers) are sending out packets. These router nodes

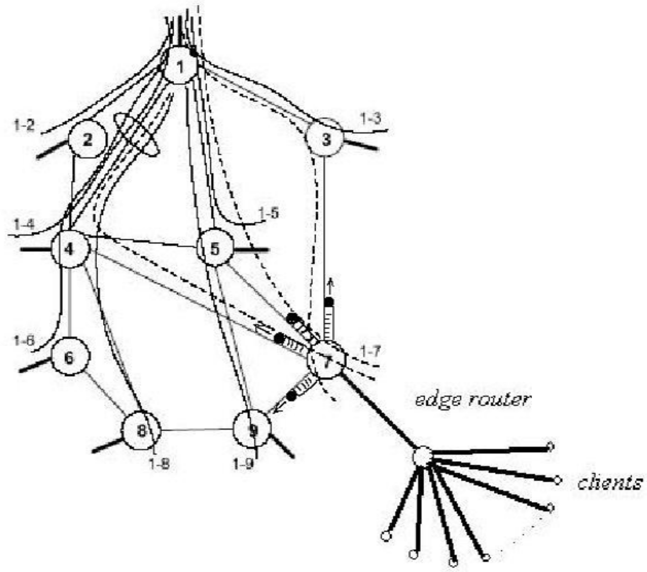


Figure 7.6: The physical scenario of the medium-scale network [81].

usually have more than one link connected to the rest of the nodes, thus they would have more than one output queue. Each link of a backbone node may have one output queue. Before the packet from a local sub-network enters the core network, they have to go through an edge router. These edge routers identify where the traffic is going to, classify it, and guide it to enter different queues of the core router nodes. For example, in the figure 7.6, node 7 has 4 output links, which are $7-3$, $7-4$, $7-5$, $7-9$. When the traffic comes from local clients, it should first go through an edge router and then distribute into one of the four forth-going queues. In order to describe the performance of the queues, the physical scenario of medium-scale network is represented in figure 7.7.

In figure 7.7, each link has a direction. The purple nodes are the core-router nodes, the yellow nodes represent the output queues for each core router, and pink nodes are source or sinks of traffic. Let us have a look at node 1. It has four queue nodes (yellow), i.e. node 10, 11, 12, 13, which are output queues for different directions of traffic flows. For example, the connection between node 1 and 2 is bidirectional, and that is represented by two directional links in the physical scenario. Yellow queue node 10 is the output queue for traffic traversing from node 1 to 2, while yellow queue node 14 for traffic going out from node 2 to node 1. The pink node 40 is the sink and source of traffic for node 1.

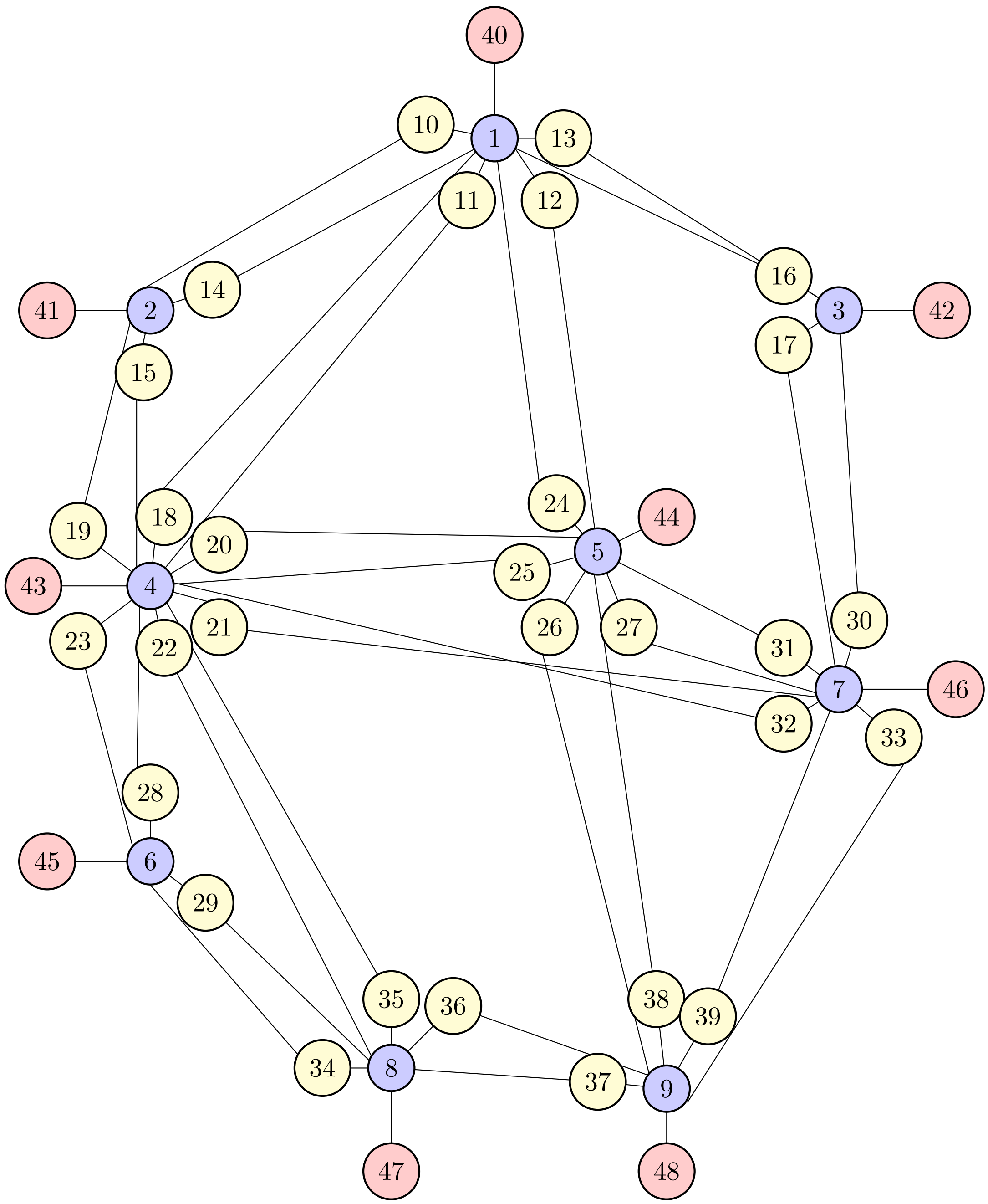


Figure 7.7: Physical representation of medium-scale network scenario

7.4.2 Results for physical scenario

Case 1 **- Deforestation algorithm with link betweenness centrality**

Figure 7.8 show the skeleton network of the physical scenario where the link utilisation is approximated using the link betweenness centrality. The algorithm groups the core

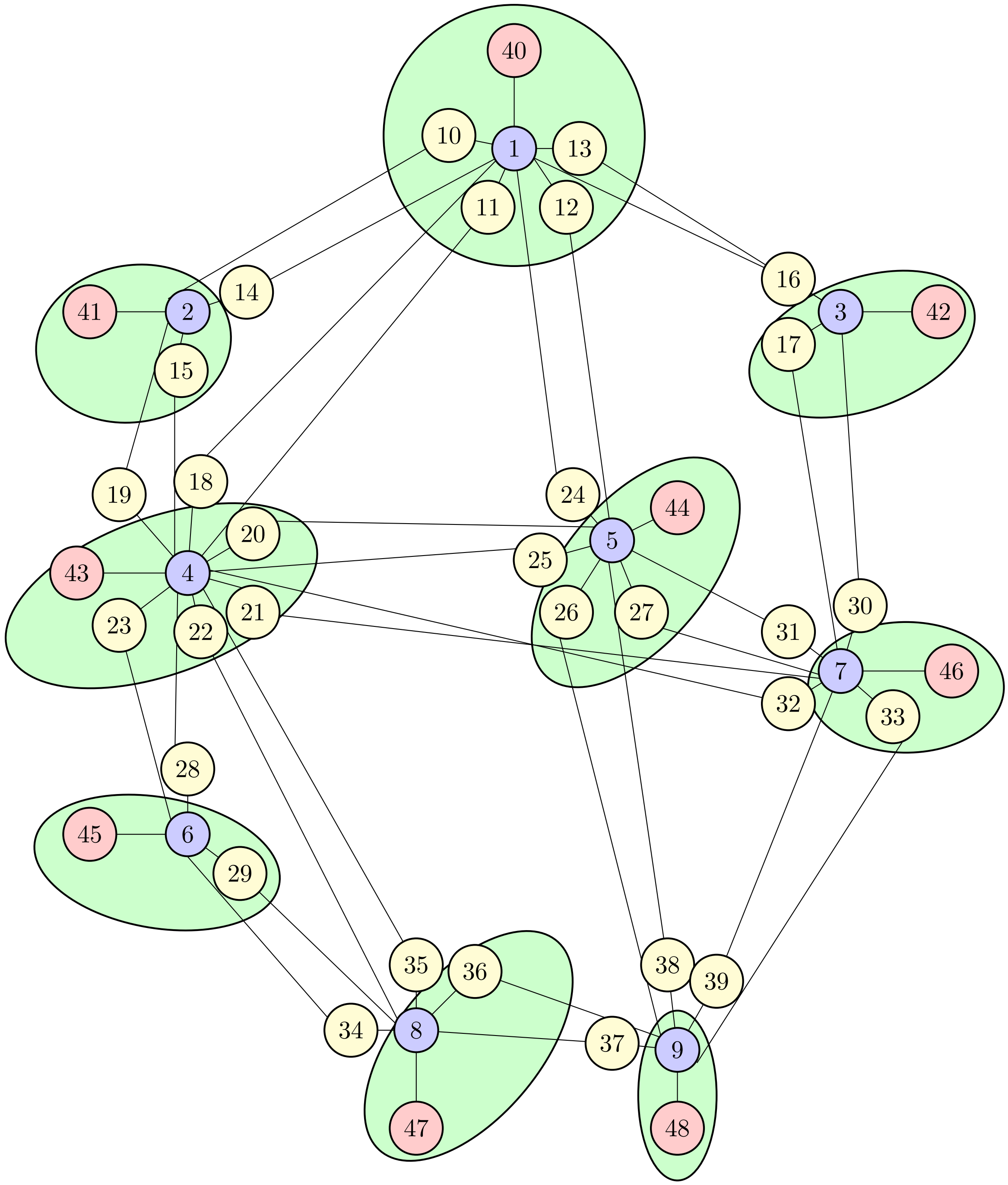


Figure 7.8: The skeleton network of the physical scenario obtained by *deforestation* (case 1).

router node, the corresponding source/sink nodes and queue nodes. In this respect, the skeleton of the physical scenario is similar to the logical scenario. Notice that this extended *deforestation* algorithm leaves some of the busy queue nodes outside the super-nodes rather than collecting them into the super-nodes, reflecting the importance of these busy queues when doing the contraction.

Case 2 - Deforestation algorithm with traffic demand matrix

In the physical representation of network, the nodes numbered from 1 to 9 are core router nodes, which could be considered as traffic sources, while from the node 40 to node 48 are sinks where the traffic terminates. Thus there are 81 ($9 * 9$) pairs of source and destination. As the physical scenario is directional, we apply an asymmetrical matrix ($TDMatrix_3$) to the network. In $TDMatrix_3$, the amount of traffic imposed at each source-destination pair is proportional to its node source number, i.e. if the packets sent by source 1 is assumed as a unit, then source i would sent i times that unit to the nine sink nodes. For example the traffic amount sent by the source node 3 to the destination node 46 is 3 units, so that the traffic imposed on the directed links between node 3 and 46 is 3 units. Whereas the traffic amount for the source-destination pair $7 - 42$ is 7 and also 7 units for the links between them.

$$TDMatrix_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 9 & 9 & 9 & 9 & 9 & 9 & 9 & 9 & 9 \end{pmatrix}$$

In this case, the skeleton network produced by *deforestation* with $TDMatrix_3$ is very similar to the skeleton network obtained in case 1 (see figure 7.8). The skeleton network contracts first the light-loaded queue nodes, and leaves the busiest queue nodes outside the super-nodes.

Generally, the *deforestation* algorithms do not work effectively for the skeleton network of the physical scenario, and they always produce the skeleton networks very similar to the logical representation of the medium-scale network. That is because the physical representation looks more complicated than the logical scenario, and it already has many triangles in the network, thus the network can not be further simplified.

7.5 Results analysis and comparison

From the previous two sections, we applied the *deforestation* algorithms on both the logical scenario and the physical scenario throughout different cases. The *deforestation* may return various simplifications depending on different selected starting nodes. The link betweenness centrality approximates the traffic status on that link, busy or not busy. This measure is used to guide the node contraction, so that high-utilised links are contracted after the low-utilised links. The simplification tends to produce a more balanced traffic distribution in super-nodes. The network traffic demands can have a great impact on the order of the contraction.

We compared our results with Qiang Yang's work [81], who also did the research in the same medium-scale scenario. Basically, his experiments are carried out using simulation tools (i.e. TOTEM toolbox), with a uniform traffic demand matrix based on the OSPF routing matrix. Thus, the link utilisation matrix obtained in his experiments are all asymmetric. That is because if there is more than one shortest-path in the network, the OSPF routing chooses only one of them as the default path. In our experiment, we applied the symmetric traffic demand matrix to the undirected logical scenario and the asymmetric matrix to the directed physical scenario. Both Qiang's and our experiments showed the same busy nodes and high-utilised links. In addition to that, the betweenness centrality of queue nodes in the physical scenario approximate how the traffic distributed to those queue nodes, and how busy the individual queue node is, which is similar to the queues' behaviour in Qiang's work.

7.6 Conclusion

To summarise, the *deforestation* algorithm can simplify the network with its connectivity only, and also can consider traffic directions and traffic demands. These traffic properties can change the order of the node contraction. The algorithm works well for both the logical scenario of networks and the physical scenario of networks. The experiments prove that the traffic dynamics on the network can have a great impact on the network simplification, hence we can conclude that a good approximation of the traffic through the network is the fundamental of a good network simplification.

Chapter 8

Network Communities

When simplifying a very large network, its skeleton network can also be large, so it is desirable to split the skeleton network into different communities. As discussed in chapter 2, many large-scale networks seem to have a *community structure*, a community of network nodes are more densely interconnected with each other than with the rest of the network, see figure 8.1. The ability to find and analyse such communities can provide invaluable help in understanding and visualizing the structure of a network.

8.1 Definition of community

The study of community structure in networks has a long history. It is closely related to the ideas of graph partitioning in graph theory, computer science and hierarchical clustering in sociology [28, 69]. There are many different definitions of what is a community, here we follow the definition by Radicchi *et. al* [61] of a weak community. A set of nodes in the network form a community if the sum of all the links inside the community is larger than the links connecting from outside to the community, i.e.

$$\sum_{i \in N} k_i^{in}(N) > \sum_{i \in N} k_i^{out}(N) \quad \forall i \in S, \tag{8.1}$$

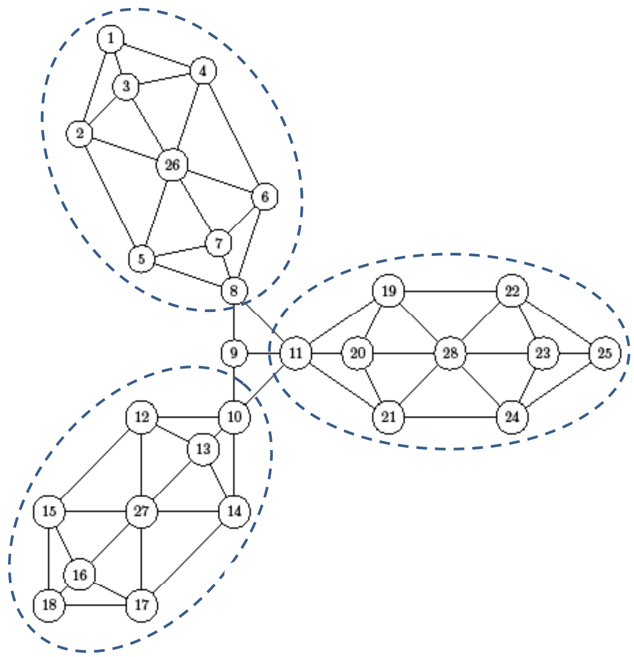


Figure 8.1: A small network with community structure. In this case there are three communities, denoted by the dashed circles, which have dense internal links but between which there is only a lower density of external links.

where S is a subgraph of a network and N is a finite set of nodes of subgraph S. k_i^{in} is the number of connections for node n_i connecting to the nodes inside while k_i^{out} is the number of connection for n_i connecting to the nodes outside the community. Community in strong sense [61] means that in the subgraph S,

$$k_i^{in}(N) > k_i^{out}(N) \quad \forall i \in S. \tag{8.2}$$

Clearly a community in a strong sense is also a community in a weak sense, while the converse is not true. Several other possible definitions are described in [75]. In general, finding an exact solution to find the community structure is believed to be an NP-hard problem, making it prohibitively difficult to solve exactly for large networks, but a wide variety of heuristic algorithms have been developed. The choice of the best method to be used depends on the configuration of the problem and the kind of desired result.

8.2 Previous work

Our work is based upon two approaches, one introduce in 1977 by Sangiovanni *et. al* [68], and the other proposed by Radicchi *et al.* in [61].

The network is described by a graph $G = (N, L)$ which consists of a set of nodes $N = \{n_1, n_2, \cdots, n_i, \cdots, n_N,\ i \in N\}$ and a set of links $L = \{l_1, l_2, \cdots, l_i, \cdots, l_L,\ i \in L\}$. To discover a community the nodes are divided into three different sets. An iterating set $\mathcal{I}$, an adjacent set $\mathcal{A}$ and the rest of the nodes $\mathcal{X}$, so that $\mathcal{X} = N - \mathcal{I} - \mathcal{A}$. The adjacent set contains all the nodes that are neighbours of the iterating set. If the adjacent set is removed then the network splits into two disconnected parts, the iterating set and the rest of the nodes.

The search for a possible community starts by putting one node in the iterating set and all its neighbours in the adjacent set. The community algorithm defines a local rule to choose which node from the adjacent set should be moved to the iterating set and also when the iterating set is a community.

In Sangiovanni *et. al* algorithm the community is defined via the number of nodes in the adjacent set, i.e. $|\mathcal{A}|$. The main step is to move one node from the adjacent set to the iterating set and re-evaluate the adjacent set. If the iterating step is labelled by the index i then the node selected from $\mathcal{A}(i)$ is the one that yields the smallest $|\mathcal{A}(i+1)|$. This last step is repeated until the size of the adjacent set goes through a minimum, at this value the iterating set forms a community. In other word the cut set $\mathcal{A}$, which separates $\mathcal{I}$ from $\mathcal{X}$, has a minimum number of nodes.

The local rule to choose which node from the adjacent set should be moved to the iterating set is a crucial point. Girvan and Newman (GN) have introduced the idea that the selection of links can be based on the value of their *link betweenness centrality* in [29]. However, the community divisive algorithm with link betweenness centrality is computationally costly, as already remarked by [56]. Evaluating the score (betweenness centrality) for all links in general requires a time $O(l * n)$, where l is the number of links and n the number of nodes.

For the worst case, the computational time is $O(l^2 n)$, which makes the analysis practically infeasible already for moderately large networks.

To overcome this problem another method is introduced which requires the consideration of local quantities only and is therefore much faster than the GN's *link-betweenness algorithm.* It is proposed by Radicchi *et al.* in [61] and based on the idea that linked nodes belonging to the same community should have a larger number of "common friends". In other words, links inside communities should be part of a large proportion of possible loops, and links connecting to nodes outside the community should have few or no loops. Instead of using the link betweenness centrality, the algorithm proceeds by using the *link clustering coefficient* - $C^{(g)}$, which represents the fraction of possible loops of order g that share a link. The algorithm is implemented for triangles ($g = 3$), see chapter 2. The algorithm computes the $C^{(g=3)}$ values for all links, and removes the ones with the *minimum* value. These two steps are repeated recursively as long as the partition fulfils the community definitions defined in Eq.(8.1).

This algorithm is very fast, since calculating the clustering coefficient can be done with local information only. This method is not appropriate for trees, sparse graphs and disassortative networks due to the small number of triangle and squares, and it also fails if the network has a small average clustering coefficient.

8.3 Finding communities via clustering coefficient

The *deforestation* generates only short cycles in the skeleton network, most of which are triangles. Hence the average clustering coefficient of skeleton network should be relatively large, so it has the right property for applying Radicchi's *clustering coefficient* division method.

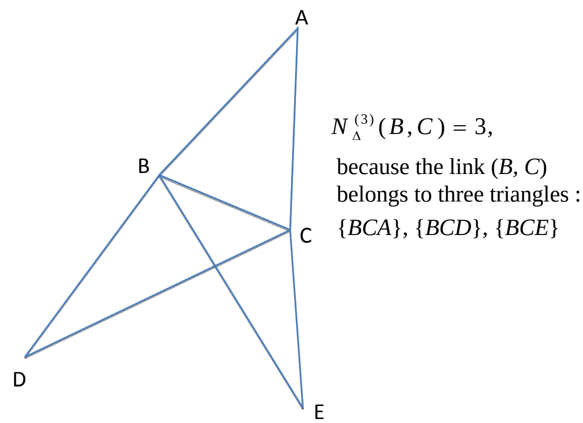


Figure 8.2: A simple example of *link clustering coefficient*.

8.3.1 Evaluation for link clustering coefficient

Recall from chapter 2, the *clustering coefficient* of link (i, j) is calculated as

$$C_{i,j}^{(3)} = \frac{N_{\triangle}^{(3)}(i, j) + 1}{min[(k_i - 1), (k_j - 1)]}, \tag{8.3}$$

where $N_{\triangle}^{(3)}(i, j)$ is the number of triangles to which $(i.j)$ belongs, and k_i, k_j are the degrees of node i, j respectively. Therefore the number of triangles belonging to a certain link (i, j) can be effectively evaluated as

$$N_{\triangle}^{(3)}(i, j) = \sum_{k \neq i, k \neq j}^{N} a_{ij} a_{ik} a_{jk}, \quad if(a_{i,j} \neq 0), \tag{8.4}$$

where a_{ij} is the element of *adjacency matrix*. A simple example of the link clustering coefficient is shown in figure 8.4. The number of triangles belonging to link (B, C) can be calculated as follows.

$$\begin{aligned} N_{\triangle}^{(3)}(B,C) &= \sum_{k\neq B, k\neq C}^{N} a_{BC}a_{Bk}a_{Ck} \\ &= a_{BC}a_{BA}a_{CA} + a_{BC}a_{BD}a_{CD} + a_{BC}a_{BE}a_{CE} \\ &= 1+1+1 \\ &= 3. \end{aligned}$$

The pseudo code for the link clustering coefficient is

Algorithm 7 Evaluate $C_{i,j}^{(3)}$

Require: Initialise a vectors $C^{(3)}$, and built adjacency matrix A
  $N_{\triangle}^{(3)}(i,j) \leftarrow 0$ {where $i, j \in N$; $N = number\ of\ nodes$}

$$a_{ij} = \begin{cases} 1 & if\ node\ i\ connects\ to\ node\ j \\ 0 & otherwise \end{cases} \tag{8.5}$$

  for $i = 1$ to N **do**
    for $j = i + 1$ to N, notice that it is $i + 1$ not 1, because A is symmetric **do**
      if $a_{ij} \neq 0$ **then**
        $N_{\triangle}^{(3)}(i,j) \leftarrow \sum_{k\neq i,k\neq j}^{N} a_{ik}a_{jk}$ {because $a_{ij} = 1$}
      end if
      Evalute k_i and k_j
      $k_i = k_j = 0$
      $k_i = \sum_{i=1}^{N} a_{ik}$; $k_j = \sum_{j=1}^{N} a_{jk}$
      $C_{ij}^{(3)} = \frac{N_{\triangle}^{(3)}(i,j)+1}{min(k_i-1,k_j-1)}$
    end for
  end for

8.3.2 Procedures for network partition

The idea behind the *link clustering coefficient* for community detection is that the links connecting nodes in different communities have few or no triangles and tend to have small values of $C_{i,j}^{(3)}$. Hence, the clustering coefficient $C_{i,j}^{(3)}$ quantities whether its link belongs to a community. The procedure for identifying network communities is shown in **Algorithm 8** (see next page).

Algorithm 8 Community detection algorithm with link clustering coefficient

Require: Network $G = (N, L)$
Require: Initialise a vector *Community*
Require: Initialise the distance matrix R from adjacency matrix A.

$$r_{ij} = \begin{cases} 1 & a_{ij} = 1 \\ \text{infinite} & a_{ij} = 0 \end{cases} \tag{8.6}$$

Evaluate the distance matrix R via *Floyd* algorithm (see appendix 2)
while There are connected links in the set L **do**
  Evaluate $C^{(3)}$
  Find the minimum value of $C^{(3)}_{ij}$, and obtain i, j
  Set $a_{ij} = 0$
  Re-calculate the distance matrix R
  if there is any $r_{ij} \to infinite$ **then**
    m++
    Move all the nodes connecting to the node i to $Community[m]$
    m++
    Move all the nodes connecting to the node j to the set $Community[m]$
  end if
end while

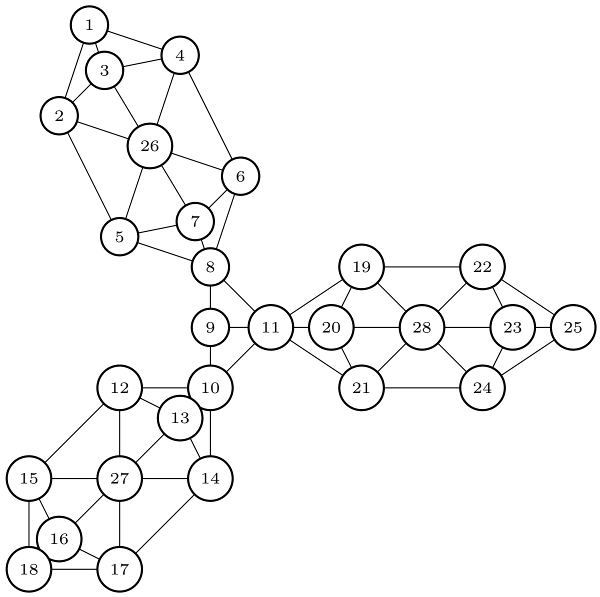
8.4 Examples

8.4.1 The community algorithm applied to a small network

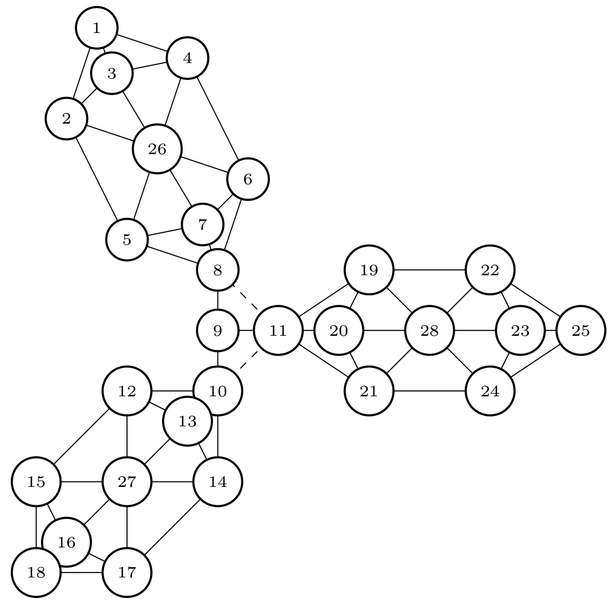
As a controlled test of how well the community algorithm performs, we carried out the experiment on a small network with a clear community structure (figure 8.1).

The whole process is completely shown in figure 8.3 (see next page). Notice that once the first link in the network is removed in such an algorithm, the clustering coefficient values for the remaining links will no longer reflect the network as it is, so the clustering coefficient needs to be re-calculated. The values of all the links are shown in the table of figure 8.3, and they are also individually updated when changed. After two rounds of removals, the network is split into 3 parts.

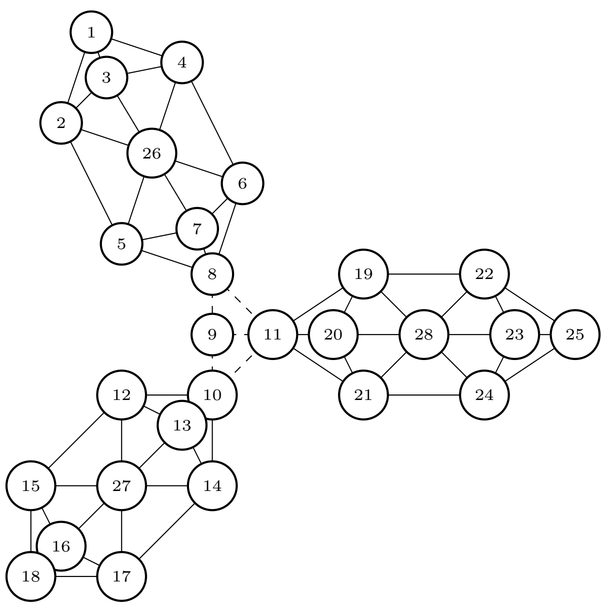
Before the removal



1^{st} removal



2^{nd} removal



Link	Clustering coefficient			Link	Clustering coefficient		
Node-Node	Before	1^{st} removal	2^{nd} removal	Node-Node	Before	1^{st} removal	2^{nd} removal
1 – 2	1.00			12 – 15	0.67		
1 – 3	1.50			12 – 27	1.00		
1 – 4	1.00			13 – 14	1.00		
2 – 3	1.00			13 – 27	1.00		
2 – 5	0.67			14 – 17	0.67		
2 – 26	1.00			14 – 27	1.00		
3 – 4	1.00			15 – 16	1.00		
3 – 26	1.00			15 – 18	1.00		
4 – 6	0.67			15 – 27	1.00		
4 – 26	1.00			16 – 18	1.50		
5 – 7	1.00			16 – 17	1.00		
5 – 8	0.67			16 – 27	1.00		
5 – 26	1.00			17 – 18	1.00		
6 – 7	1.00			17 – 27	1.00		
6 – 8	0.67			19 – 20	1.00		
6 – 26	1.00			19 – 22	0.67		
7 – 8	1.00			19 – 28	1.00		
7 – 26	1.00			20 – 21	1.00		
8 – 9	1.00	0.50	0.00	20 – 28	1.00		
8 – 11	0.50	0.00	0.00	21 – 24	0.67		
9 – 10	1.00	0.50	0.00	21 – 28	1.00		
9 – 11	1.50	0.50	0.00	22 – 23	1.00		
10 – 11	0.50	0.00	0.00	22 – 25	1.00		
10 – 12	0.67			22 – 28	1.00		
10 – 13	1.00			23 – 24	1.00		
10 – 14	0.67			23 – 25	1.50		
11 – 19	0.67			23 – 28	1.00		
11 – 20	1.00			24 – 25	1.00		
11 – 21	0.67			24 – 28	1.00		
12 – 13	1.00						

Figure 8.3: A simple example of community algorithm based on *link clustering coefficient*.

8.4.2 Deforestation and community algorithm applied to a small network

The *deforestation* algorithm itself can divide the original network into communities. The super–nodes are communities that are defined by having only one path between the members of the community. In order to simplify more, we apply the community divisive algorithm

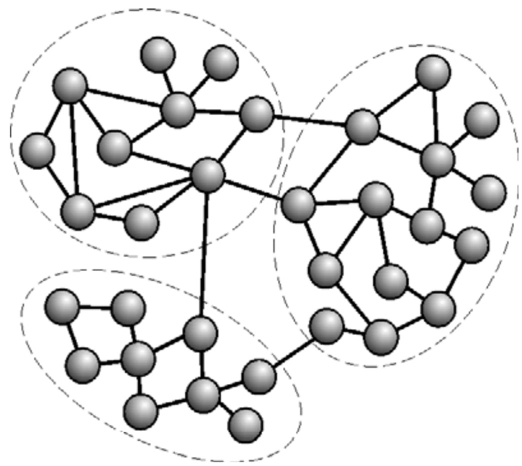


Figure 8.4: A network with community structure represented by the dashed lines. The communities are the groups of more intensely interconnected nodes. [17]

Link	**Clustering coefficient**					
Node-Node	Original	1^{st} removal	2^{nd} removal	3^{rd} removal	4^{th} removal	5^{th} removal
$1-2$	2.00					2.00
$1-3$	2.00					2.00
$2-3$	2.00					2.00
$3-4$	1.00			2.00	1.00	
$3-12$	0.67				0.00	
$4-5$	0.50			0.00		
$4-12$	1.00			2.00	1.00	
$5-6$	0.67			1.00		
$5-9$	1.00			1.50		inf
$5-10$	1.00					
$6-7$	2.00					inf
$6-8$	1.00					
$6-9$	1.00					
$7-8$	2.00					inf
$8-9$	1.00					
$8-10$	1.00					
$8-11$	0.67	0.33	0.00			
$8-12$	0.40	0.00				
$9-10$	1.50					inf
$11-12$	1.00	0.67	1.00			
$11-13$	0.67		1.00			
$11-14$	1.50					inf
$12-14$	1.00					
$12-15$	2.00				1.00	
$12-16$	0.67				0.00	
$13-14$	1.00					
$13-16$	0.67				0.00	
$13-17$	2.00				1.00	
$15-16$	2.00				1.00	
$16-17$	2.00				1.00	

Table 8-1: The clustering coefficient value for each link are recorded and correspondingly updated after the removal every time. And here the "inf" means the Infinite.

to the skeleton network obtained by *deforestation*. In the following, we applied both *deforestation* algorithm and Radicchi *et al.* method to a simple network, which has a known community structure delimited by the dashed line (see figure 8.4).

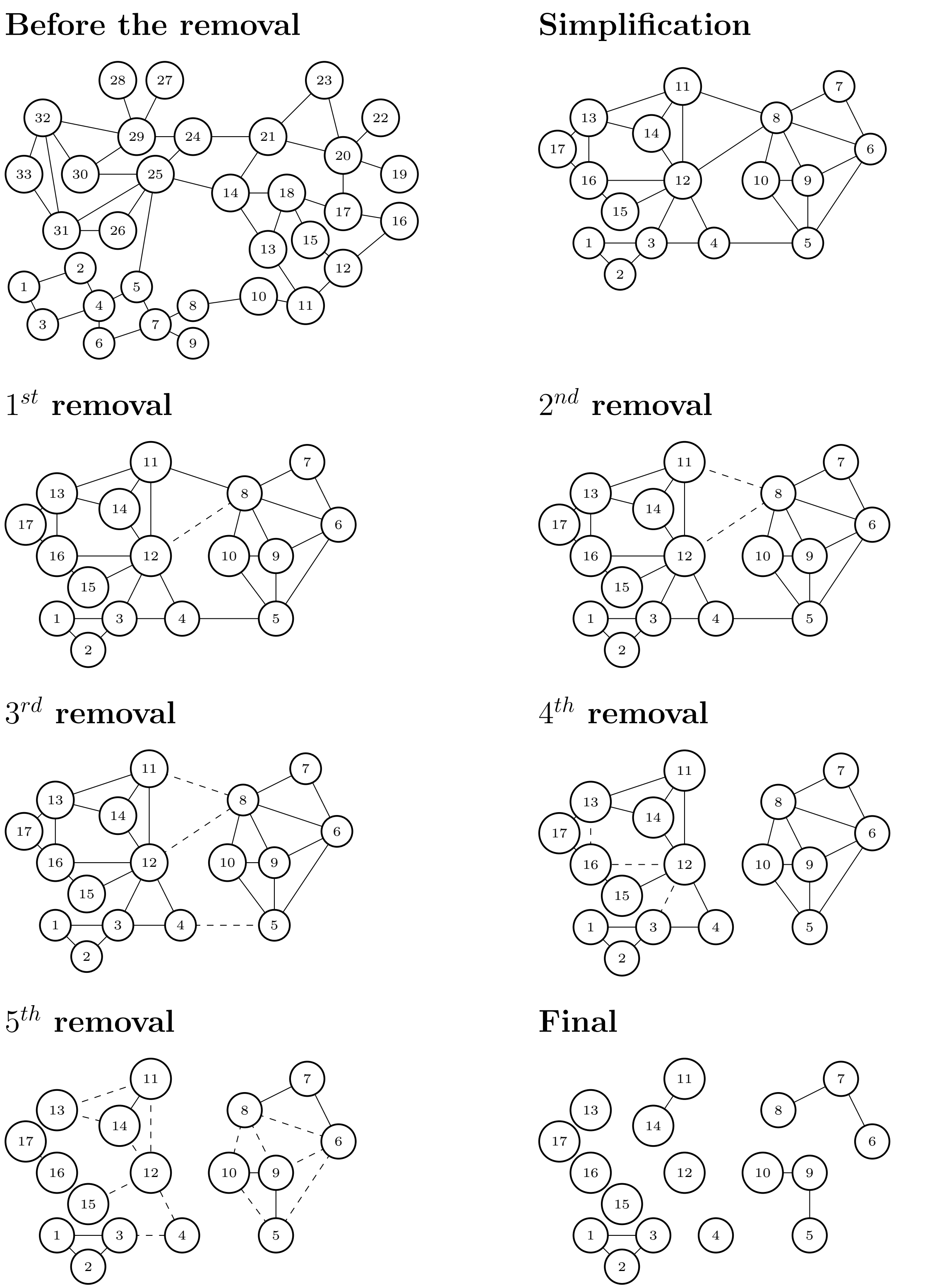


Figure 8.5: Algorithms of *deforestation* and Radicch *et. al.* division method detect the community structure for a simple network.

Figure 8.5 illustrates the whole process that the network was simplified and was finally split by progressively removing the links. After the 3rd round of the removal, the network is splitting into 2 parts, one of which is matched to one of three communities shown in the original network (see figure 8.4). And after the 5th removal, the network seems to be split into a number of small fractions, i.e. isolated nodes and pieces of still connected subnetworks. The only triangle left, is the one that was composed of super–nodes $1, 2$ and 3, and which actually contain the original nodes $1, 2, 3, 4$ and 6. So this fraction of the triangle also corresponds to one of the three communities of the original network. This example demonstrates that *deforestation* and Radicchi *et al.* method can work together to divide the network into communities. Although some communities were extracted at the final step, it can show the major division of the network at the first few steps. Table $8 - 1$ recorded and updated the clustering coefficient value for all the links during the whole experiment.

8.4.3 Internet AS network & IPA networks

In practical situations the algorithms will normally be used on networks for which the communities are not known ahead of time. The correct partition of the network sometimes can be found at the early stages of the Radicchi's algorithm and sometimes at the last stage. However we believe that methods like the one presented here can reduce the network complexity, and are invaluable to understand the structure of large-scale networks.

Applying the Radicchi *et. al* method to reveal the community structure of the Internet AS–skeleton network (the one in chapter 4), we noticed that the AS–skeleton network forms a tight community. The continuous application of Radicchi *et. al* method removes the peripheral nodes but there is always a core set of nodes that remain well connected. The same observation has also been found in the independent study of [3]. This finding is very important and shows that the AS network has a single core, which is tightly connected. And this core is conserved during the simplification.

The IPA–skeleton is different, as the algorithm removes some of the peripheral links the network splits into two parts. The smaller part contains 1/3 of the network nodes. This suggests that there is a bottleneck in the number of alternative routes within these two parts and there is no single core in the IPA network. It may be the result of geographic factors or by some other reasons. At the moment, we can not give an accurate reason for the existence of this bottleneck. In general, the *deforestation* and Radicchi *et. al* method together can provide a solution to not only detect the community structure but also reveal important properties of network structure during the simplification.

8.5 Conclusion

The *deforestation* algorithm produces a skeleton network which has a high average clustering coefficient, so it is appropriate to use the Radicchi *et al.* method to split the skeleton networks into communities. This method is simple and fast. The practical example of the Internet shows that the structure of AS network and IPA network are different. The AS network has a core that is well-connected and cannot be split into different communities. This reflects that there is a set of AS nodes (the core) that play a fundamental role in the functionality of the whole network. However, the IPA network can be split into two major communities. That means there are two major “address” space communities. From the data (CAIDA), we were not able to confirm if this is the case as the data has been anonymised.

Chapter 9

Network Structure Revealed by Short Cycles

In chapter 8, we have studied the community structure of the Internet. And we observed that the AS-Internet has a tightly connected core, and it is hard to simplify the core further. The core is the part of the skeleton network, which consists of cycles only. This provides us a hint of how to study the tightly-connected network from another feature of the network. Cycles has been acknowledged as particularly important for the complex network, and their statistical distribution of cycles underlies the connectivity of the networks [67], and also have an impact on the dynamics running though the networks [5]. In this chapter, we investigate the cycles of different lengths of the networks.

9.1 The importance of cycle basis

The cycle structure of networks is an old topic that has occupied electrical engineers for nearly a century, and it has recently become an attractive topic again in many real-world applications, e.g. analysis of chemical and biological pathways, periodic scheduling, graph drawing, and routing mechanisms [67].

Stable, scalable, adaptive, distributed routing schemes are important attributes for current and future communications networks. Traditionally, routing protocols represent the connectivity of a network as a small number of distinct trees, one tree for every source. This information is translated into routing table entries. This table of connectivity alone tells little about the intrinsic diversity of the network and, therefore, its resiliency to attacks or attrition. However recently, more and more new routing schemes have been proposed, i.e. the R^3 protocol[1] introduced by Alexander Stepanenko *et al.* [72], which summarises and exploits the network's potentially rich path diversity. Central to this routing scheme is the *Logical Network Abridgement* (LNA) procedure, which makes use of the concept of minimal cycle basis to complete a hierarchical topological abstraction [16]. This procedure is performed iteratively till the network can not be abstract any more. Then every level of abstraction summarises path diversity information for the previous level. Therefore, routing can take place at the LNA abstraction of all the lower level. This procedure also summarises path diversity information of fairly large networks in a scalable fashion and can be augmented with a number of forwarding rules to create a resilient recursive routing (R^3) protocol.

It is also possible to simplify the skeleton network by dividing it into different communities by using the cycle basis. James Bagrow *et la.* [6] apply short cycle detection, e.g. minimum cycle basis, to help to identify communities in complex networks. In terms of communities, most inter-community links contain few (if any) short cycles, but intra-community links tend to contain both long and short cycles, since a long cycle can coil inside the community.

The network cycles can also aid visualisation of the topological structure of large-scale networks, and they may give a better understanding the general organization and involvement of such complex structures.

[1] The R^3 protocol (Resilient Recursive Routing) is a dynamic link-state protocol, providing a scalable routing solution that is capable of automatically optimising performance of highly dynamic traffic flows in large, packet-switched connectionless networks. Such traffic flows could be associated with large numbers of broadband hosts on a public network, large numbers of enterprise hosts on a private network, or large numbers of military hosts on a defence network.

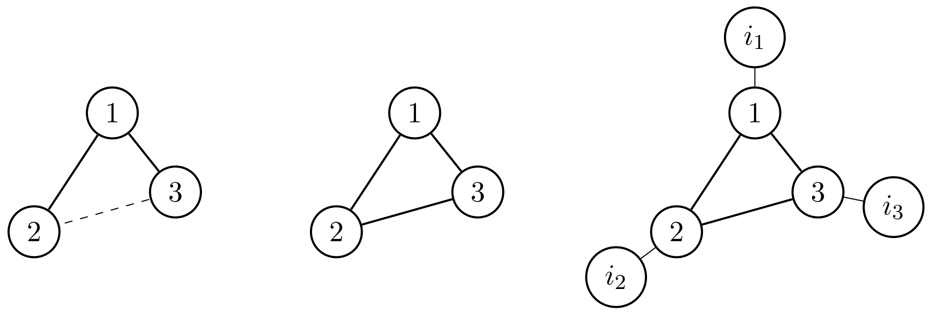


Figure 9.1: The basic cycle of size 3.

9.2 Transitive connectivity and resilience

In communications networks, the fundamental property is path diversity, which is strongly related to the existence of cycles. The cycle of nodes is simplest and smallest diversity topological unit, and it affords two path choices to go from any node in the cycle to any other node in the same cycle.

Cycles such as the basic cycle of size 3 in figure 9.1 occur sooner or later along the network evolution. For instance, indirect information exchange between nodes 2 and 3 (i.e. through node 1) is likely to foster the appearance of the direct link between those two nodes. In other words, the formation of such cycles can be understood as a *reinforcement* of the connectivity between the involved nodes, possibly implied by intensive information interchange. Therefore, the density of cycles (such as 3-cycle) is likely to provide interesting insights about the growth dynamics and connectivity properties of complex networks. Another important aspect intrinsic to the cycles is the *transitivity* of connections along the network. In other words, if node 1 is connected to node i_1, node 2 is connected to node i_2 and node 3 is connected to node i_3, the eventual presence of virtual link (path) is established extending direct from node i_1 to nodes i_2 and i_3, which can be understood as an indication of transitivity in the network connectivity.

Without path diversity there are no routing decisions to be made and adaptivity via routing becomes irrelevant. Routing must exploit path diversity to achieve network resilience to congestion and link or node failures. The density of cycles of different lengths can be used as an indicator of path diversity. In other words, the larger the number of shortest cycles among a subset of nodes, the more connected such nodes are to one another. Therefore the number of alternative paths in the network gives an idea of the network robustness to failure.

The collection of all cycles in a network can form a vector space [32], called the *cycle space* Z_m (see the definition in chapter 2). A cycle basis of the graph G is defined as a basis for the cycle space. Any cycle Z can be expressed as $\sum_{i=1}^{M(G)} Z_i$ where $Z_1, Z_2, Z_3, \ldots, Z_{M(G)}$ form the cycle basis. The number of cycles in the cycle basis, or cyclomatic number M. The cycle basis is a compact description of the set of *independent* cycles that suffice in describing the cycle structure of a network. In previous chapters, we know the deforestation algorithms simplify all the trees in the network, and conserve the same number of cycles in the skeleton network; the majority of which are short cycles, from which the cycle basis can be easily extracted.

In this chapter, we start by presenting an algorithm for finding the minimum cycle basis, and then show its application in the very large complex networks.

9.3 Minimum cycle basis algorithm

As the deforestation process does not change the number of cycles, the cyclomatic number of the original graph and and the skeleton graph are the same. The skeleton network has the same number of alternative paths as the original network, however, the length of the paths are different.

9.3.1 The algorithm

The problem of computing a minimum cycle basis (MCB) in an undirected network graph has been extensively studied [36]. Here we briefly describe and implement an efficient algorithm proposed by Kurt Mehlhorn *et al.* [46]

The links of network graph G have non-negative weights. The weights of a cycle is the sum of the weights of its links, and the weight of a cycle basis is the sum of the weights of its cycles. We simply consider each link has the same weight, which equals to 1. Let T be any spanning forest of G, and let $l_1, \cdots, l_M$ be the links of $G \setminus T$ in some arbitrary but fixed order. Note that the cyclomatic number M is exactly the dimension of the cycle space.

The algorithm below computes the cycles of an MCB and their *witness*. A witness $\mathcal{S}$ of a cycle Z is a subset of $l_1, \cdots, l_M$ which will prove that Z belongs to the MCB. Both cycles and witnesses are vectors in the space $0, 1$. $< Z, \mathcal{S} >$ stands for the standard inner product of vectors Z and $\mathcal{S}$. We observe that $< Z, \mathcal{S} >= 1$ if and only if the cardinality of the intersection of the two link set is odd. Finally, adding two vectors Z and $\mathcal{S}$ is the same as the symmetric difference of the two link sets. The algorithm gives a full description, and the symbol $\bigoplus$ denotes the symmetric difference[2]. The algorithm in phase i has two parts, one is the computation of the cycle Z_i and the second part is the update of the sets $\mathcal{S}_j$ for $j > i$. Note that updating the sets $\mathcal{S}_j$ for $j > i$ is nothing more than maintaining a basis $\mathcal{S}_{i+1}, \cdots, \mathcal{S}_M$ of the subspace orthogonal to $Z_1, \cdots, Z_i$.

Algorithm 9 Construct a MCB

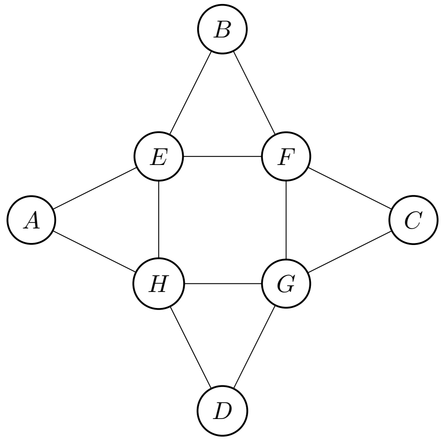
Require: Set $\mathcal{S}_i = l_i$ for all $i = 1, \cdots, M, and\ M = cyclomaticnumber$

```
for i = 1 to M do
  Find Z_i as the shortest cycle in S s.t. < Z_i, S_i >= 1
  for j = i + 1 to M do
    if < Z_i, S_j >= 1 then
      S_j = S_j ⊕ S_i
    end if
  end for
end for
```

9.3.2 Examples

Figure 9.2 is an example of a small network consisting of cycles of different lengths. Intuitively, it has 4 triangles (3-cycles), 1 quadrangle (4-cycle) and couples of 5-cycles, 6-cycles, etc, all of which form the cycle space for the network. We also observe that the cyclomatic

[2]In mathematics, the *symmetric difference of two sets is the set of elements which are in one of the sets, but not in both. This operation is the set-theoretic kin of the exclusive disjunction (XOR operation) in Boolean logic.*



Link	Node - Node
l_1	$A-B$
l_2	$A-H$
l_3	$B-E$
l_4	$E-F$
l_5	$E-H$
l_6	$B-F$
l_7	$F-C$
l_8	$F-G$
l_9	$C-G$
l_{10}	$D-G$
l_{11}	$G-H$
l_{12}	$D-H$

	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9	l_{10}	l_{11}	l_{12}
$\mathcal{S}_1$					1							
$\mathcal{S}_2$						1						
$\mathcal{S}_3$								1				
~~$\mathcal{S}_4$~~									~~1~~			
$\mathcal{S}_5$										1		
${\mathcal{S}_4}^*$								1	1			
$\mathcal{Z}_1$	1	1			1							
$\mathcal{Z}_2$			1	1		1						
$\mathcal{Z}_3$							1	1	1			
$\mathcal{Z}_4$										1	1	1
$\mathcal{Z}_5$				1	1			1			1	

Figure 9.2: The vectors of the cycle basis and the witness for the network in the figure. And ${\mathcal{S}_4}^*$ is an update witness for $\mathcal{S}_4$.

number of the network is 5, as $M(G) = L - N + K = 12 - 8 + 1 = 5$. Hence the cycle basis of this network is composed of 5 independent simplest cycles, which are from the cycle space and also can be used to describe all the other cycles in the cycle space. The table in figure 9.2 demonstrates the process of constructing the minimum cycle basis. The witnesses can be obtained from the minimum spanning tree algorithm. Every witness belongs to one of cycles in cycle basis, so 5 witnesses are corresponding to 5 cycles in the cycle basis separately. In the table, ${\mathcal{S}_4}^*$ is an update witness for $\mathcal{S}_4$, where ${\mathcal{S}_4}^*$ is orthogonal to the accepted witnesses. Hence, from the table, the cycle basis $Z =< Z_1, Z_2, Z_3, Z_4, Z_5 >$ of the network in figure 9.2 are the cycles $< AEH >$, $< BEF >$, $< CFG >$, $< DGH >$ and $< EFGH >$.

9.3.3 An example of the real world network

We also computed the minimum cycle basis in the AS-Internet skeleton network. As deforestation procedure generates short cycles (i.e. all possible triangles) in the AS skeleton network, from which the cycle basis can be easily extracted. Among 12236 cycles in the cycle basis, there are 11516 triangles (94.12%) and 179 quadrangles, which shows the transitivity of AS skeleton network is very high. This shows that the AS skeleton gives a very compact representation of the AS graph, from which the minimum cycle basis can be easily extracted, and hence many relevant applications can be applied.

9.4 Conclusion

Commonly, the structure of large complex networks is characterized using statistical measures. These measures can give a good description of the network connectivity but they do not explore the interaction between the dynamical process and network connectivity. Deforestation algorithms produce the skeleton network consisting of short cycles only, which provides us a different way of studying network structure and dynamics. The short cycles existing in the skeleton network can be used as an indicator of network transitivity and resiliency, and also can easily form a set of independent simplest cycles, called the cycle basis. This fundamental diversity space motivates a wide range of application in the context of telecommunication networks: A novel resilient recursive routing (R^3) protocol, visualisation of a hierarchical topological abstraction of large networks, and the methods of community structure identification.

Chapter 10

Conclusion & Future Work

This thesis has presented a novel method, the *deforestation* algorithm, to simplify very large networks. The *deforestation* is general and it conserves the same number of alternative paths between all the sources and destinations, hence that the the network's resilience (path diversity) is retained. However the *deforestation* does not give a unique simplification of the network; it depends on the order in which the nodes were contracted. We have taken advantage of this property and extended the deforestation algorithm to include several restrictions when doing the simplification, for example the direction of the links and the flow through the links.

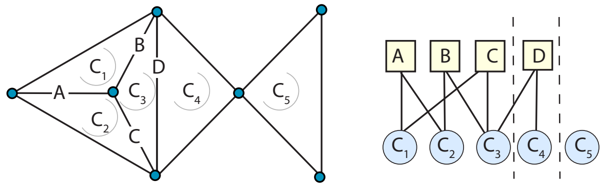
In the thesis we also show that the properties of the original network, like the density of connections, determines the properties of the super-nodes and the skeleton network. Once the deforestation is applied, other methods developed for complex network can be used in the skeleton network to obtain relevant information. The deforestation process divides the original network into communities, which are defined by having only one path between the members of the community. However, the skeleton networks can also be large, so it is desirable to simplify the skeleton networks further to split into different communities. In this case a community is defined as a collection of nodes that are more densely connected than expected. We show that the splitting of the skeleton network reflect the characteristics that there are relatively few alternative paths between communities. Other property that can be easily extracted from the skeleton network is the cycle basis, which can suffice in describing the cycle structure of complex network. This cycle basis has been used for adaptive routing (R^3 protocol) and visualisation of large complex graphs.

We have tested the new algorithm, the *deforestation* algorithm on the Autonomous System (AS) level and Internet Protocol address (IPA) level of the Internet. For both the AS and IPA network, the skeleton graph has approximately 50% fewer nodes than the original network. The AS skeleton network inherits the characteristics of its core due to the correlation of the interconnectivity of the high degree nodes. However, for the IPA network the high degree nodes are sparsely connected in contrast with its skeleton network which are tightly connected. We also notice that if we contracted the AS network considering the weights of the links, we obtain a more balanced reduction of the network, where the weights between the super–nodes tend to be more homogeneous. From the community detection algorithm we find that the skeleton of the AS network forms a tight community and always has a core set of nodes that remain well connected under continuous removal. However the skeleton of the IPA network is different; it is split into two parts. This suggests that there is a bottleneck in the number of alternative routes within IPA network.

We also show that *deforestation* algorithm can be used to simplify a medium scale network with traffic directions and traffic demand matrix. The network simplification takes into consideration the traffic constraints, and that some nodes represent sources or sinks of traffic and other nodes as queues.

Based on the above, the deforestation algorithm is general, also simple, fast and flexible. It is a promising method for analysing large-scale complex network, and has a wide range of applications and extensions.

relationship between cycles as a bipartite graph



Further simplification

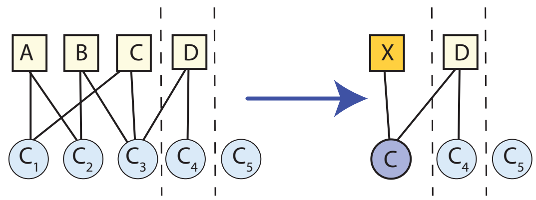


Figure 10.1: The relationship between the cycles expanding the network and their common links (left) are represented with a bipartite graph (right).

10.1 Further work

10.1.1 Using cycle basis to detect network hierarchies

. One aspect of this work that could be extended is to divide the simplified network into different hierarchies by using the cycle basis. Consider all the cycles of the basis and the links that are shared by these cycles. Build a new graph where there are two kind of nodes, the cycles of the original network and links shared by these cycles. A link in this new graph represents that at least two cycles share a link of the original networks (see figure 10.1). This gives a bipartite graph which relates cycles and links. This bipartite graph can be split into hierarchies depending on the connectivity of the nodes and the weights of the nodes representing the links of the original network. In figure 10.1 circuit C_5 can be split from the rest of the network as it does not shares a link with the other cycles. The rest of the network can be considered as divided into two hierarchies, cycles C_1, C_2, and C_3 belong to one hierarchy defined by a tighter connectivity between the cycles. Cycle C_4 is weakly connected to the other cycles (also see figure 10.1).

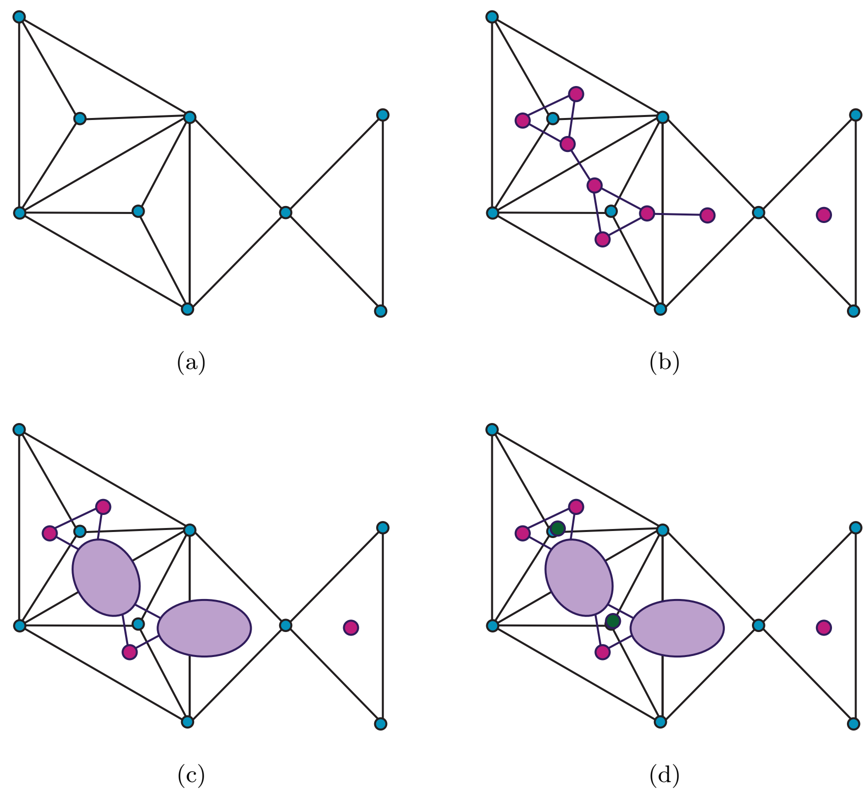


Figure 10.2: Abstraction of cycle of cycles into elementary units of diversity. And a simple network is applied by the abstraction and *deforestation* recursively till no more cycles can be abstracted.

10.1.2 Cycle of cycles for visualisation and routing mechanism.

The skeleton network can determine a cycle basis (see chapter 9), e.g. minimal cycle basis. We can abstract all cycles in the basis as a dot node and join the dot nodes where adjacent cycles share a common link. An abstraction of the skeleton network has obtained (see figure 10.2). Then we apply the *deforestation* algorithm and the abstraction method recursively till no cycles can be abstracted. An example of a simple network is shown in figure 10.2. From the figure, we noticed that every level of abstraction summarises path diversity information for the previous level, and the abstraction of cycle of cycles can be aggregated into elementary units of diversity. It is beneficial for visualisation very large networks and also agglomeration path diversity for routing at each level.

Appendix A

Graph Isomorphism

Given two networks, it is difficult and also interesting to discover if they are same or not. In graph theory, it is called *isomorphism*. The exact isomorphism and the sub-graph isomorphism detection play a key role and are used in a variety of real applications, such as chemistry, information retrieval, networking and linguistics, etc. Two isomorphic graphs must have exactly the same set of parameters as given in figure $A.2$. In figure $A.1$, two graphs, $G = (N, L)$ and $H = (V, E)$, are *isomorphic* (normally written in the form $G = H$) if there are bijections $g : N \to V$ and $h : L \to E$ [62, 65].

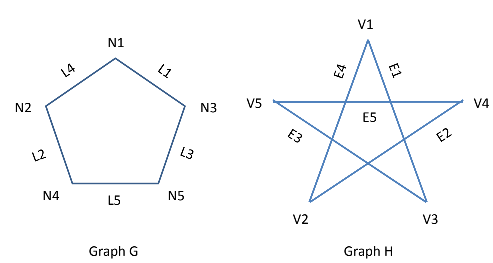


Figure A.1: A pair of isomorphic graphs

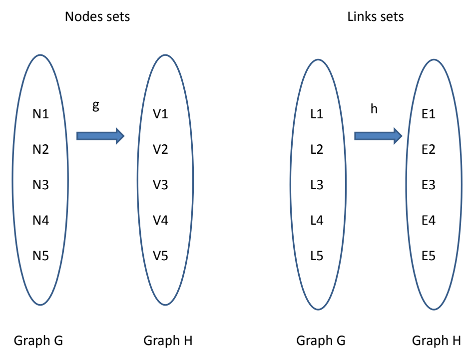


Figure A.2: One to one correspondence of nodes and links for graph G and H

We often identify that two graphs are not isomorphic by showing that invariants under isomorphism. The invariants used for comparison are as follows:

1. Number of nodes should be same in both matrices.

2. Number of links in both matrices should be equal.

3. In both of the matrices the nodes having same degree are grouped to form classes. The number of classes should be equal.

4. Total degree of matrices should be same.

After the above invariants are checked, if any of these quantities differ in two graphs, then the two graphs are not isomorphic. However, when these invariants are the same, it does not necessarily means that the two graphs are isomorphic. It is often difficult to determine whether two simple graphs are isomorphic or not. There are $N!$ possible one-to-one correspondences between the node sets of two simple graphs with n nodes. Testing each such correspondence to see whether it preserves adjacency and non-adjacency is impractical if N is large. Thus an algorithm guaranteeing a solution in running time proportional to a constant power of N - the number of nodes, is desirable, but no such algorithm has been discovered for determining if two arbitrary graphs are isomorphic.

Appendix B

Floyd Algorithm

The **Floyd algorithm** (also variously known as **Floyd-Warshall** algorithm, the Roy-Floyd algorithm, or the WFI algorithm) is named after Robert Floyd and Stephen Warshall [24]; is an algorithm for efficiently and simultaneously find the shortest paths (i.e. graph geodesics) between every pair of nodes in a weighted and potentially directed graph. A single execution of the algorithm will find the shortest paths between all pairs of nodes.

Given a graph $G = (N, L)$, which comprises a set of N nodes $\{n_i\}$, and a set of $L \subseteq N * N$ links connecting nodes in N. In a directed graph, each link also has a direction, so links (n_i, n_j) and (n_j, n_i), $j \neq i$, are distinct. A graph can be represented as an adjacency matrix A in which each element (i, j) represents the link between element i and j. A_{ij} if there is an link (n_i, n_j); otherwise, $A_{ij} = 0$. A *path* from node n_i to node n_j is a sequence of links $(n_i, n_k), (n_k, n_m), \cdots, (n_t, n_j)$ from L in which no nodes appears more than once.

Algorithm. Floyd's all-pair shortest-path algorithm is given as below, see pseudo code. It take as input an $N * N$ adjacency matrix A and compute an $N * N$ matrix D, with D_{ij} the length of the shortest path from n_i to n_j, or distinguished value (∞) if there is no path.

The algorithm derives the matrix D in N steps, constructing at each step k an intermediate matrix $I(k)$ containing the best-known shortest distance between each pair of nodes. Initially, each $I_{ij}(0)$ is set to the length of the link (n_i, n_j), if the link exists, and to ∞

Algorithm 10 Floyd(G)

Require: Nodes N, Links L

```
I_ij(0) ← 0 {if i = j}
I_ij(0) ← length((n_i, n_j)) {if link exists and i ≠ j}
I_ij(0) ← ∞ {otherwise}
for k = 0 to N − 1 do
  for i = 0 to N − 1 do
    for j = 0 to N − 1 do
      I_ij(k + 1) = min(I_ij(k), I_ik(k) + I_kj(k))
    end for
  end for
end for
D=I(N)
```

otherwise. The kth step of the algorithm considers each I_{ij} in turn and determines whether the best-known path from n_i to n_j is longer than the combined lengths of the best-known paths from n_i to n_k and from n_k to n_j. If so, the entry I_{ij} is updated to reflected the shortest path. This comparison operation is performed a total of N^3 times.

References

[1] R. Albert and A. L. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1), 2002.

[2] R. Albert, H. Jeong, and A. L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, July 2000.

[3] J. I. Alvarez-Hamelin, L. D. Asta, A. Barrat, and A. Vespignani. k-core decomposition: a tool for the analysis of large scale internet graphs. *CoRR*, abs/cs/0511007, 2005.

[4] J. I. Alvarez-Hamelin, L. D. Asta, A. Barrat, and A. Vespignani. K-core decomposition: a tool for the visualization of large scale networks. *CoRR*, abs/cs/0504107, 2005.

[5] A. Arenas and P. C. J. Guilera, D. A.and Vicente. Synchronization reveals topological scales in complex networks. *Phys Rev Lett*, 96(11), March 2006.

[6] J. Bagrow, E. Bollt, and L. F. Costa. Network structure revealed by short cycles, 2006.

[7] V. Batagelj and A. Mrvar. *Pajek - analysis and visualization of large networks*. Springer, 2003.

[8] M. Baur, U. Brandes, M. Gaertler, and D. Wagner. Drawing the as graph in 2.5 dimensions. In *Graph Drawing*, pages 43–48, 2004.

[9] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[10] CAIDA. Cooperative association for internet data analysis. http://www.caida.org/home.

[11] D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Network robustness and fragility: Percolation on random graphs. *Physical Review Letters*, 85:5468, 2000.

[12] K. Calvert, E. Zegura, and J. Sterbenz. Canes: A modest spproach to active networking. *IEEE Comm. Mag.*, 35:160–163, 1997.

[13] Q. Chen, H. Chang, R. Govindan, and S. Jamin. The origin of power laws in internet topologies revisited. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 608–617 vol.2, 2002.

[14] R. Cohen, K. Erez, D. Avraham, and S. Havlin. Resilience of the internet to random breakdowns. *Phys.Rev.Lett*, 85:4626, 2000.

[15] V. Colizza, A. Flammini, M. A. Serrano, and A. Vespignani. Detecting rich-club

ordering in complex networks. *Nature Physics*, 2:110, 2006.

[16] L. F. Costa and et. al. *Handbook of Applied Algorithms: Solving Scientific, Engineering, and Practical Problems*. Wiley, 2008.

[17] L. F. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas. Characterization of complex networks: A survey of measurements. *Advances In Physics*, 56:167, 2007.

[18] Lazowska E. D., J. Zahorjan, Graham S. G., and K. C. Sevcik. *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*. Prentice-Hall, 1984.

[19] L. Danon, J. Duch, A. Arenas, and A. Diaz-Guilera. Community structure identification. Physical Review Letters, Oct 2005.

[20] S. N. Dorogovtsev and J. F. F. Mendes. *Evolution of networks: From biological nets to the Internet and WWW*. Oxford University Press, Oxford, January 2003.

[21] P. Erdos and A. Renyi. On random graphs. *Publications Mathematicae*, 6:290, 1959.

[22] P. Erdos and A. Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5, 1960.

[23] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM*, 1999.

[24] R. Floyd and W. Robert. Algorithm 97: Shortest path. *Communications of the ACM*, 1962.

[25] L. C. Freeman, S. P. Borgatti, and D. R. White. Centrality in valued graphs: A measure of betweenness based on networks flow. *Elsevier Science Publishers*, pages 141–154, 1991.

[26] M. L. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(35):35–41, 1977.

[27] H. Fuks and A. T. Lawniczak. Performance of data networks with random links. *Mathematics and Computers in Simulation*, 51:103–119, 1999.

[28] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W. H. Freeman and co., 1979.

[29] M Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc.Natl.Acad.Sci.USA*, 99:7821, 2002.

[30] A. Gunnar, M. Johansson, and T. Telkamp. Traffic matrix estimation on a large

ip backbone - a comparison on real data. In *ACM 1-58113-821-0/04/0010 IMC'04 Taormina, Sicily, Italy*, October 25-27, 2004.

[31] B. Halabi. *Internet Routing Architectures*. Cisco Press, 1997.

[32] F. Harary. *Graph Theory*. Addison Wesley, 1969.

[33] I. Herman, G. Melancon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on visulization and computer graphics*, 6(7), 2001.

[34] P. Huang, D. Estrin, and J. Heidemann. Enabling large–scale simulations: Selective abstraction approach to the study of multicast protocols. *Int. Symp. on Modelling Analysis and Simulation of Computer and Telecommunication Systems, Montreal Canada*, 1998.

[35] M. Junger and P. Mutzel. *Graph Drawing Software*. Springer, 2004.

[36] T. Kavitha, C. Liebchen, K. Mehlhorn, D. Michail, R. Rizzi, T. Ueckerdt, and K. A. Zweigk. Cycle bases in graphs characterization, algorithms, complexity, and applications. *Computer Science Review*, 3(4):199–243, 2009.

[37] A. Kershenbaum. *Telecommunications network design algorithms*. McGRAW-Hill, Inc., 1993.

[38] W. H. Kim and T. W. Chien. *Topological analysis and synthesis of communication networks*. Columbia University Press, 1962.

[39] L. Kleinrock. *Queueing systems*. Wiley Interscience, 1975.

[40] L. Liu and R. J. Mondragón. A novel algorithm for simplifying networks. In *PGNET-6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, 2005.

[41] L. Liu and R. J. Mondragón. Simplification of large networks using conservation of alternative paths. In *Proceedings of the 2008 Networking and Electronic Commerce Research Conference*, 2008.

[42] L. Liu and R. J. Mondragón. Simplifying large complex networks via conserving path diversity. In *NetSci'08 International Workshop and Conference on Network Science and Its Applications*, 2008.

[43] L. Liu and R. J. Mondragón. Conservation of alternative paths as a method to simplify large networks. In *SIMPLEX '09: Proceedings of the 1st Annual Workshop*

on Simplifying Complex Network for Practitioners, pages 1–6, New York, NY, USA, 2009. ACM.

[44] T. J. McCABE. A complexity measure. *IEEE Transactions on software engineering*, se-2, No. 4:308–320, 1976.

[45] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic matrix estimation: Existing techniques and new directions. In *SIGCOMM'02*, August 19-23, 2002.

[46] K. Mehlhorn and D. Michail. Implementing minimum cycle basis algorithms. *J. Exp. Algorithmics*, 11:2.5, 2006.

[47] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[48] OSI model. The open system interconnection reference model (osi reference model or osi model) is an abstract description for layered communications and computer network protocol design. http://www.automatedbuildings.com/news/oct06/reviews/OSI.gif.

[49] R. J. Mondragón and S. Zhou. Ensembles related to the rich-club coefficient for non-evolving networks. *arXiv/0808.4062*, 2008.

[50] M. E. J. Newman. Assortative mixing in networks. *Physical Review Letters*, 89:208701, 2002.

[51] M. E. J. Newman. A measure of betweenness centrality based on random walks, September 2003.

[52] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.

[53] M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B - Condensed Matter*, 38(2):321–330, March 2004.

[54] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 74(3), 2006.

[55] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, June 2006.

[56] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.

[57] J. Park and M. E. J. Newman. The origin of degree correlations in the internet and

other networks. *Physical Review E*, 68:026112, 2003.

[58] R. Pastor-Satorras, A. Vázquez, and A. Vespignani. Dynamical and correlation properties of the internet. *Physical Review Letters*, 87(25):258701+, November 2001.

[59] V. Paxon and S. Floyd. Why we don't know how to simulate the internet. In *In proceedings of the 1997 Winter Simulation Conference*, pages 1037–1044, 1997.

[60] V. Paxson and S. Floyd. Why we don't know how to simulate the internet. pages 1037–1044, 1997.

[61] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proc Natl Acad Sci U S A*, 101(9):2658–2663, March 2004.

[62] V. Remie. Bachelors project: Graph isomorphism problem. Master's thesis, Eindhoven University of Technology, Department of Industrial Applied Mathematics, September 5, 2003.

[63] DTI Innovation Report. Competing in the global economy: the innovation challenge, December 2003.

[64] Report from Internet World Stats. Miniwatts Marketing Group Report. *World Internet Users and Population Stats*, June 2009.

[65] K. Riaz, M. Sikander, H. Khiyal, and M. Arshad. Matrix equality: An application of graph isomorphism. *Information Tehnology Journal*, 4(1):6–10, 2005.

[66] G. Riley and M. Ammar. Simulating large networks: How big is big enough? In *Proceedings of First International Conference on Grand Challenges for Modeling and Simulation*, Jan. 2002.

[67] H. D. Rozenfeld, J. E. Kirk, E. M. Bollt, and D. B. Avraham. Statistics of cycles: How loopy is your network? *J.PHYS.A*, 38:4589, 2005.

[68] A. Sangiovanni-Vincentelli, L. Chen, and L O. Chua. An efficient heuristic cluster algorithm for tearing large-scale networks. *IEEE Transactions on Circuits and Systems*, CAS-24(12), 1977.

[69] J. Scott. *Social nework analysis: A handbook*. Sage Publications, 2000.

[70] W. D. Seary, A. J. andRichards. Partitioning networks by eigenvectors. In *In Proceedings of the International Conference on Social Networks*, volume 1, 1995.

[71] S. B. Seidman. Network structure and minimum degree. *Social Networks*, 5:269–287,

1983.

[72] A. S. Stepanenko and C. C. Costas. Novel topological framework for adaptive routing. In *Proceedings of 3rd International Conference on the Latest Advances in Networks (ICLAN 2008)*, 2008.

[73] J. W. Stewart. *BGP4: Inter-domain routing in the Internet*. Addison-Wesley, 1998.

[74] C. Walshaw. A multilevel algorithm for the force-directed graph drawing. *Graph Drawing Symposium*, pages 171–182, 2000.

[75] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge Univ. Press, Cambridge, U.K., 1994.

[76] D. J. Watts. *Small worlds*. Princeton University Press, 2003.

[77] D. J. Watts and S. H. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393:440–442, 1998.

[78] B. M. Waxman. Routing of multipoint connections. *IEEE J. Select. Areas Commun.*, 6(9):1617–1622, 1988.

[79] W. Willinger, R. Govindan, S. Jamin, V. Paxon, and S. Shenker. Scaling phenomena in the internet: Critically examining criticality. *Proc. Natl. Acad. Sci. USA 99*, pages 2573–2580, 2002.

[80] ITU X.200. Information technology - open systems interconnection - basic reference model: The basic model. http://www.itu.int/rec/T-REC-X.200-199407-I/en.

[81] Q. Yang. *Scalable quality of service in converged IP networks*. PhD thesis, Queen Mary, University of London, 14 May 2007.

[82] L. Zhao, Y.-C. Lai, K. Park, and N. Ye. Onset of traffic congestion in complex networks. *The American Physical Society*, 71(2):026125(8), 2005.

[83] S. Zhou and R. J. Mondragón. The rich-club phenomenon in the internet topology. *Communications Letters, IEEE*, 8(3):180–182, 2004.

Author's Publications

1. L.Liu, and Raúl J. Mondragón, A Novel Algorithm for Simplifying Networks, *PGNET-6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking & Broadcasting, Liverpool*, UK, 2005

2. L.Liu, and Raúl J. Mondragón, Simplifying Large Complex Networks via Conserving Path Diversity, *NetSci'08 International Workshop and Conference on Network Science and Its Applications*, UK, 2008

3. L.Liu, and Raúl J. Mondragón, Simplification of Large Networks using Conservation of Alternative Paths, *Proceedings of the 2008 Networking and Electronic Commerce Research Conference*, Italy, 2008

4. L.Liu, and Raúl J. Mondragón, Conservation of alternative paths as a method to simplify large networks, *In Proceedings of the 1st Annual Workshop on Simplifying Complex Network For Practitioners*, SIMPLEX '09 ACM, Venice Italy, July 01 2009