Network Information Flow for multicast communication networks

Project report submitted in partial fulfillment

of the requirements for the degree of

MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Nikhil Bhargava (2004MCS2650)

M-TECH Student

IIT Delhi

INDIAN INSTITUTE OF TECHNOLOGY, DELHI

New Delhi - 110016, India

July 13, 2009

Certificate

This is to certify that the project report titled "**Network Information Flow for multicast communication networks**" being submitted by Nikhil Bhargava (2004MCS2650) for the partial fulfillment of the requirements for the degree of Master of Technology is a bonafide work carried out by him under my guidance and supervision. The work presented in this report has not been submitted elsewhere, either in part or full, for the award of any degree.

Dr. S.N Maheshwari
(Project Supervisor)
Department of Computer Science and Engineering
Indian Institute of Technology - Delhi
New Delhi, India.

Acknowledgement

I would like to thank both my project advisors, Dr. S.N Maheshwari, Computer Science and Engineering, IIT Delhi, and Dr. Naresh Sharma, School of Technology and Computer Science, TIFR (Mumbai) for their guidance, time and expertise that they have provided through out my study. My research work would not have been completed satisfactorily without their help. I would also like to thank Mr Jaswant Singh, GCL-II lab in-charge, for providing lab facilities as and when required. Finally, I would thank all my friends and family members who have supported me throughout this work.

Nikhil Bhargava
2004MCS2650

Abstract

Increasing throughput and maximizing bandwidth usage is a potential requirement in both unicast and multicast communication networks. For any given point-to-point communication network with multiple mutually exclusive information sources multicasting information bits to some sets of destinations, it is difficult to characterize the admissible coding rate region. Network switching alone can never achieve max-flow min-cut bound; however by employing coding at the nodes, referred to as network coding, bandwidth, in general, can be saved. Ahlswede, Cai, Li, and Yeung [1] have shown that maximum achievable information rate by network coding in a single source multicast network is more than that for the case of network switching. We have this problem of finding the maximum admissible coding rate region for a given network and then devising a suitable information multicast strategy for a given communication network. We have analyzed the switching gap for a special network, defined as the ratio of maximum achievable information rate using network coding (NC) to that of network switching (NS). In this work, we have found the switching gap of a singular-symmetric, dual-symmetric, triple-symmetric and then give an intuitive observation for the n^{th} version of singular symmetric butterfly network which we term as generic butterfly network. We have also done an extended survey of the previous work done in area of network switching and explained major results.

Keywords: Network switching, Network coding, max-flow min-cut theorem, multicast networks, convex Optimizations, Linear Programming, Route packing

Contents

List of Figures

Chapter 1

Introduction

In their pioneering work on network coding [1], Ahslwede et al. determined the capacity for multicasting information in a network of lossless channels. Specifically, the network is represented as $G = (V, E, c)$, where (V, E) is a directed graph and c is the edge capacity vector of length $|E|$. Ahslwede et al. have established a theory of network coding for single-source and multisource-source multicast networks. For single-source network coding, they demonstrated that the maximum achievable information rate for network coding is always upper bounded by max-flow min-cut bound. Further, they demonstrated by many examples that this bound cannot be achieved by conventional network switching and some kind of network coding [3] has to be applied at each node of the network. Li, Yeung, and Cai [2] showed that the multicast capacity can be achieved by linear network coding. Since past 9 years, recent discoveries in this field have generated a lot of interest in research fraternity. A comprehensive survey on the theory of network coding is presented in Yeung, Li, Cai, and Zhang in [5].

Network switching and network coding are two techniques of data combination to increase throughput although the former is a special case of the latter. Switching gap or coding gain [6], is defined as the ratio of maximum achievable information rate using network coding (NC) to that of network switching (NS). A natural problem arises from here, what is the switching gap for a given single source multicast network and under what conditions does

switching gap reaches its maxima and minima (which will be 1). Xue-Bin Liang in [7] has analyzed the Ahlswede-Cai-Li-Yeung's classical butterfly network [1] to determine switching gap of the network as well as determined certain conditions on link capacities such that switching gap of the network comes out to be unity.

We have looked on the problem of network information flow in a single source multisink multicast network. Post [1] era, many people have worked on developing optimal strategies for network information flow for different class of networks. We have tried to extend Liang's work [7] by taking modified versions of butterfly network and finding its switching gap under a given set of conditions on the link capacities of the network. In this process, we analyze singular-symmetric, dual-symmetric and triple-symmetric butterfly network and then give an expression for a special case of a generic butterfly network.

This chapter gives a short introduction to the problem. The rest of the report is organized as follows. Chapter 2 gives motivation for this problem as well as the formal problem statement. Chapter 3 gives a detailed description of network coding. Chapter 4 gives a detailed description on network switching. Chapter 5 gives analysis of singular-symmetric butterfly network, dual-symmetric butterfly network, triple-symmetric butterfly network and finally generic butterfly network. Chapter 6 concludes the study.

Chapter 2

Motivation and Problem Statement

A lot of research in the realm of network coding have happened in recent years which provides the motivation for our work. Let $G = (V, E, c)$ a point-to-point communication network represented by a directed graph, where V is the finite set of vertices in the network and E is the finite set of edges connecting two vertices say u and $v \in V$ in the network G and c is the capacity function for E. We do not consider any edge connecting a node to itself i.e., in other words we are not considering multigraph. Further we rule out cycles in the graph which essentially means that we consider the case of acyclic networks only.

Each edge or link, say e is associated with a capacity c which belongs to set of positive rational numbers, R^+. This network could be used to transmit information from one node to other in the network. We can safely assume that transmission in the network will be error free if and only if, transmission rate, r over any link e should be lesser then link capacity, c. Let X be the information source, generating information in bit sequences spread over the field Ψ at node $s \in V$ in the network G. Information gets transmitted from s to every destination or sink nodes, t_1, $t_2, \ldots$, $t_L \in V$ such that information could be reconstructed at each of t_i. Hence, without loss of any generality, we can say that information from source X gets multicast to L distinct sinks in G. This boils down to a single source-multisink problem in a multicast point to multi point communication network. Please note that for both Network switching (NS) and Network coding (NC), there will not be any information loss, however

flow might not be conserved.

An interesting problem which is only partially explored is that, when is information rate achieved by network coding (NC) equal to information rate achieved by (NS) under a given set of conditions on link capacities. We have tried to solve this problem for a specific case that could be considered as a generic version of Ahlswede-Cai-Li-Yeung's butterfly network after modifying this network by taking capacities of all links at same level to be equal (see chapter 5 for further details). We have then tried to find the maximum transmission rate or the switching gain for generic single source multi sink network with unit capacity edges. Although it is a known hard problem [8, 28], but we were able to achieve some bounds on the rate.

Chapter 3

Network coding

This chapter gives a detailed description of the field of network coding as well as summarizes pioneering work done in this field.

3.1 Introduction to Network Coding

Network coding is a new research area that have interesting applications in practical networking systems. With network coding, intermediate nodes may send out packets that are linear combinations of previously received information. There are two main benefits of this approach i.e. potential throughput improvements and a high degree of robustness.

In network coding, a node acts as encoder/decoder. Instead of simply forwarding data, nodes may recombine several input packets into one or several output packets. A simple example in a wireless context is a three node topology, as shown in figure 3.1.

Here, say A and B are wireless nodes (transceivers) and want to exchange packets via an intermediate node S (say, wireless base station). A [resp. B] sends a packet a [resp. b] to B, which then broadcasts a xor b instead of a and b in sequence. Both A and B can recover the packet of interest, while the number of transmissions is reduced by 1 and bandwidth is

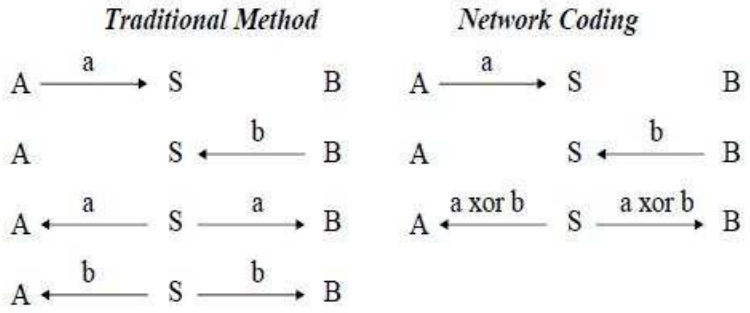

Figure 3.1: Example of Network coding.

conserved. Linear network coding replaces the xor operation is by a linear combination of the data, interpreted as numbers over some pre defined finite field. Linear combining requires enhanced computational power at the nodes of the network. But nowadays, processing is becoming less and less expensive. Thanks to Moores law, the bottleneck has shifted to network bandwidth in order to support the ever-growing demand in applications and QoS guarantees over large unreliable networks. Network coding utilizes cheap computational power to increase network efficacy.

Consider a system that acts as information relay, such as a router, a node in an ad-hoc network, or a node in a peer to peer distribution network. Traditionally, when forwarding an information packet destined to some other node, it simply repeats it. With network coding, we allow the node to combine a number of packets it has received and transmit into one or several outgoing packets. Assume that each packet consists of L bits. We can interpret s consecutive bits of a packet as a symbol over the field $\mathbb{F}_{2^s}$ with each packet consisting of a vector of $\frac{L}{s}$ symbol. Linear combination is not concatenation by any means. If we linearly combine packets of length L, the resulting encoded packet also has size L. With linear network coding, outgoing packets are linear combinations of the original packets and encoding and decoding algorithms are not only easy to understand and implement but can be done in polynomial time.

3.2 Process of Network Coding

Assume that a number of original packets $M^1, \ldots, M^n$ are generated by one or several sources. In linear network coding, each packet in the network is associated with a sequence of coefficients $g_1, \ldots, g_n$ in $\mathbb{F}_{2^s}$ and is equal to $X = \sum_{i=1}^{n} g_i M^i$. The summation has to occur for every symbol position, i.e., $X_k = \sum_{i=1}^{n} g_i M_k^i$, where M_k^i and X_k is the kth symbol of M^i and X respectively.

Lets say a packet contains both the coefficients $g = (g_1, \ldots, g_n)$, called encoding vector, and the encoded data $X = \sum_{i=1}^{n} g_i M_i$, called information vector [3]. The encoding vector is used by recipients to decode the data. For example, the encoding vector $e_i = (0, \ldots, 0, 1, 0, \ldots 0)$, where the 1 is at the ith position, means that the information vector is equal to M^i (i.e., is not encoded). At some special intermediate nodes and all sink nodes, decoding of received packets is done. Assume a node has received the set $(g^1, X^1), \ldots, (g^m, X^m)$. In order to retrieve the original packets, it needs to solve the system $\{ X^j = \sum_{i=1}^{n} g_i^j M^i \}$ (where the unknowns are M^i). This is a linear system with m equations and n unknowns. However the condition $m \geq n$ is not sufficient, as some of the combinations might be linearly dependent. Hence it is essential to choose the optimal linear codes for each node.

Past research has shown mainly two approaches to choose the linear codes. The first approach is to choose random codes. A simple random algorithm for choosing network code is to have each node in the network select uniformly at random the coefficients over the field $\mathbb{F}_{2^s}$ [4]. However this approach is independent and decentralized. The second approach is to use deterministic algorithms for network coding. The polynomial-time algorithm for multicasting in [16] sequentially examines each node of the network, and decides what linear combinations each node performs. Since each node uses fixed linear coefficients, the packets only need to carry the information vector. There also exist deterministic decentralized algorithms that apply to restricted families of network configurations [17].

3.3 Benefits of Network Coding

Network Coding can increase the capacity of a network for multicast flows. To make this point more clear, consider a network that can be represented as a directed graph. The vertices of the graph correspond to terminals, and the edges of the graph corresponds to channels. Assume that we have M sources, each sending information at some given rate, and N receivers. The case when N receivers share the network resources, each of them can receive the maximum rate it could hope to receive, even if it were using all the network resources by itself.

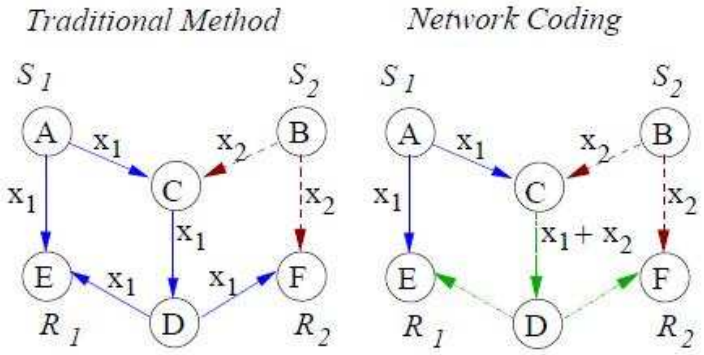

Figure 3.2: Data transmission in a butterfly network.

Figure 3.2 show a Butterfly Network with two sources S_1 and S_2 multicast to both R_1 and R_2. All links have unit capacity. Only edge CD is shared between the two paths. With network coding (by xoring the data on link CD), the maximum achievable rate is 2 for each source, the same as if every destination were using the network for its sole use. With network switching, the achievable rates are less. In this case considering both rates equal, network switching can achieve a maximum rate of 1.5. This holds for multicasting. Consider the case of unicast traffic. In figure 3.2, say source S_1 transmits to destination R_2 and S_2 to R_1. Network coding can achieve a rate of 1 for each receiver, while without that, we can only send data at rate $\frac{1}{2}$ to each receiver. Further, network coding allows to achieve the optimal throughput when multicasting using polynomial time algorithms.

The most compelling benefits of network coding might be in terms of robustness and adapt-

ability. Consider again figure 3.1 and assume that A and B may go into sleep mode (or may move out of range) at random and without notifying the base station S. If the base station S broadcasts a (or b), the transmission might be completely wasted, since the intended destination might not be able to receive. However, if the base station broadcasts a xor b, or random linear combinations of the information packets, the transmission will bring new information to all active nodes.

Network coding can offer benefits for delay sensitive and high data rate applications in networks where packets get dropped. Two main approaches are employed today: Automatic repeat request (ARQ) schemes that achieve the optimal rate at the cost of delay; and packet-level forward error correcting (FEC) schemes that achieve the optimal delay at the cost of rate. For example, consider a source A that would like to transmit information to a destination C. On the path from A to C there exists a router B that can perform network coding operations. Assume that node A sends encoded packets, that are dropped on paths AB and BC with probability ϵ_{AB} and ϵ_{BC} respectively. Letting destination C decode the packets it receives, restricts the rate to $R_1 \leq (1-\epsilon_{AB})(1-\epsilon_{BC})$. If we allow the router B to perfectly decode and re-encode, we will achieve the optimal min-cut rate $R_2 \leq \min\{(1-\epsilon_{AB}), (1-\epsilon_{BC})\}$, but at the cost of additional delay: we have to wait at node B to receive sufficient encoded packets to be able to decode and re-encode the information. Using an ARQ scheme will again allow to achieve rate R_2, but again at the cost of increased delay. Alternatively, node B can, at each time instance, form and send random linear combinations of the encoded packets it has received up to that time, without waiting for all encoded packets. We can then achieve the optimal rate R_1 without an additional delay.

3.4 Application of Network coding

The most widely known application using network coding is Avalanche [18, 19]. Generally, in a peer-to- peer content distribution network, a server splits a large file into a number of blocks. Peer nodes try to retrieve the original file by downloading blocks from the server but also distributing downloaded blocks among them. To this end, peers maintain connections

to a limited number of neighbouring peers (randomly selected among the set of peers) with which they exchange blocks. In Avalanche, the blocks sent out by the server are random linear combinations of all original blocks. A node can either determine how many innovative blocks it can transmit to a neighbor by comparing its own and the neighbors matrix of decoding coefficients, or it can simply transmit coded blocks until the neighbor receives the first non-innovative block. The node then stops transmitting to this neighbor until it receives further innovative blocks from other nodes. Network coding minimizes the download times. With network coding, the performance of the system depends much less on the specific overlay topology and schedule. Due to the diversity of the coded blocks, a network coding based solution is much more robust in case the server leaves early or in high churn rates (where nodes only join for a short period).

Another application area of network coding is transmission in wireless networks. Network coding can improve throughput in multihop routing in wireless networks with bidirectional routing with same number of packets to exchange. Given a schedule that alternates between adjacent routers, after a few initial steps, all inter- mediate routers have packets buffered for transmission in both directions of the path. Whenever a transmission opportunity arises, a router combines two packets, one for each direction, with a simple xor and broadcasts it to its neighbors. Both receiving routers already know one of the packets the broadcast is coded over, while the other packet is new. Thus, each broadcast allows two routers to receive a new packet, effectively doubling the capacity of the path. Further, simple xor based network coding can increase throughput in residential wireless mesh networks [20].

Network coding also finds application in adhoc networks and sensor networks. [21] describes a novel and interesting application for network coding with untuned transceivers in sensor networks [36]. Another interesting data gathering algorithm for sensor networks is presented in [22]. Network coding also finds application in the areas of network tomography and network security. [23] describes an application of network coding in inferring the loss rates of links in an overlay network. For conventional active probing, packets are usually multicast to several receivers. The receivers experience the same loss event which provides information about losses in the underlying multicast tree. After a sufficiently large number of probe

packets, shared links and their loss rates can be identified with reasonable accuracy. In such a setting, network coding provides additional flexibility since packets are not only duplicated at branching points of the multicast tree, but may also be merged. If multiple senders unicast packets to a single receiver, and these packets are combined within the network, it allows to infer the topology in much the same way as multicasting from one sender to multiple receivers. Furthermore, if the network code (i.e., the specific way in which packets are combined at the nodes) is known in advance, the coding coefficients contained in the probe packets provide additional information about the original packets that were combined (and consequently which packets were lost in which part of the tree). By exploiting these features, it is possible to significantly reduce the number of active probes and the link stress generated by these probes. Network coding has also been proposed for passive network monitoring.

Network coding also finds application in network security where in it helps in protection from a potential eavesdropper, since information is more spread out and thus more difficult to *overhear*. In [24], the authors investigate the problem of designing secure network codes for wiretap networks, where certain known links are tapped by attackers. The source combines the original data with random information and designs a network code in such a way that only the receivers are able to decode the original packets. Furthermore, the mutual information between the packets obtained by the eavesdroppers and the original packets is zero. The authors, K. Bhattad and K. R. Narayanan, in [25] investigate a weaker form of security, based on the fact that nodes can only decode packets if they have received a sufficient number of linearly independent information vectors, which an eavesdropper might not be able to do. Network coding also simplifies the protection against modified packets in a network [26]. In a network with no additional protection, an intermediate attacker may make arbitrary modifications to a packet to achieve a certain reaction at the attacked destination. However, in the case of network coding, an attacker cannot control the outcome of the decoding process at the destination, without knowing all other coded packets the destination will receive.

3.5 Previous work done

The problem of computing network information flow in a network introduced in [1] triggered the question that how much information can be transmitted through a network. The field of network coding has roots in information theory as well as connections to two classic graph problems i.e. Steiner tree packing and multicommodity flow.

Information theorists have concentrated on communication over a single channel. [27] describes a set of senders who wants to transmit information across a channel to a set of receivers. Mote specifically, it introduces the problem of network information flow as given many senders and receivers and a channel transition matrix which describes the effect of the interference and noise in the network, decide whether or not the sources can be transmitted over the channel or not. This problem involves distributed source coding as well as distributed communication. This work has started research on data compression and error-tolerance. In networks it is unlikely and tough to model a network as a single channel. For example, in a network communication problem, different receivers may have access to different channels in a network. In this case, it is multi channel communication problem. Characterizing the capacity of an information network has remained an open problem for decades.

Network coding is a variation of max flow concept in graph theory. However information flow is different from fluid flow. Network coding does not follow the principle of conservation of flow but obey capacity constraint. In a traditional view of communication in a network, data can be replicated at nodes but not encoded together with other data. The problem of packing fractional Steiner trees in a graph can be used to model this type of communication. The Steiner packing problem is to find the maximum number of edge-disjoint subgraphs of a given graph G that connect a given set of required points S [28]. Given a directed capacitated graph G, a node r and a subset of vertices S, a fractional Steiner tree packing is a set of fractional Steiner trees such that for every edge in G the total weight of trees containing that edge is no more than the capacity of the edge. The objective of the fractional Steiner tree packing problem is to maximize the total weight of the set of Steiner trees. The problem

of multicasting from a source s to a set of sinks $t_1, \ldots, t_n$ has traditionally been studied as a fractional Steiner tree packing problem. Each tree in a fractional Steiner packing can be used to send a fraction of the data as per its weight. Computing optimal fractional Steiner packing in undirected graphs is NP hard [28].

Another line of thought that went in to develop concepts of network coding is multi commodity flow problem in graph theory. In an instance of the multicommodity flow problem, there is a directed capacitated graph G and k commodities. For each commodity i, there is a single source s_i, a single sink t_i, and a demand d_i. In a multicommodity flow, the total flow of all commodities across an edge in G must be no more than the capacity of that edge. The objective is to find the largest fraction r such that for every commodity i, at least rd_i units of commodity i flow from source s_i to sink t_i. This problem can be solved optimally using linear programming [29].

Ahlswede et al. in [1] for the very first time introduced the concept of network coding. The presented a modified form of max-flow min cut theorem for information networks and proved that a source can multicast k messages to a set of sinks provided the min-cut between the source and each sink has capacity k. Network coding was also introduced by R.J Mceliece in his talk [13] where in the author showed how multicast trees could be used to maximize data transmission in information network.

Li, Yeung and Cai showed that the maximum achievable rate for the multicast problem can always be achieved using a linear code [2]. This focused attention on using linear codes for network coding as their viability to solve a wider array of network coding problems. They also proved that linear codes could be constructed deterministically but the algorithm is exponential in size of network. Peter et al. in [39] presented a polynomial time algorithm for Network Information Flow in a graph.

Koetter and Médard devised an algebraic framework for network coding in [30]. Using the algebraic framework proposed, the authors studied the application of linear network coding to directed cyclic graphs. Ho et al. in [31] analyzed a particular class of networks to construct a randomized algorithm that can be executed in a distributive fashion. Petar et al. gave

efficient methods of linear network coding in [37].

Jaggi et al. devised a polynomial-time implementation of the Li, Yeung and Cai multicast algorithm in [32]. It gives a linear solution over a field $\mathbb{F}$ such that $|\mathbb{F}| = O(number\ of\ sinks)$. This triggered much of research work in determining whether an even smaller alphabet would suffice or not. A second important contribution of Jaggi et al. was an instance on n nodes in which network coding achieves a rate $\Omega(logn)$ times larger than the best rate achievable with fractional Steiner tree packing [28].

Li et al. [33] considered network coding in undirected graphs. They showed that fractional Steiner tree packing techniques can achieve at least $\frac{1}{2}$ the maximum possible multicast rate. This put an upper bound on the usefulness of network coding in undirected graphs for the multicast problem. Desmond et al. in [38] studied network coding in networks having certain cost to operate. They showed that the minimum-cost multicast problems without network coding are very difficult except in the special cases of unicast and broadcast, finding minimum-cost subgraphs for single multicast connections with network coding can be posed as a linear optimization problem.

Liang in [7] analyzed the butterfly network given in [1] to determine the ratio of maximum rate achieved by network coding to that of network switching. He termed this ratio is switching gap or coding gain of the network. He concluded that the switching gain would always be greater than 1 except for a special case. Chandra Chekri in [35] examined the throughput of network coding with the average throughput achievable by routing, where the average throughput refers to the average of the rates that the individual receivers experience. Chen et al. in [40] investigates the diversity gain offered by implementing network over wireless communication links. Their results show that Distributed Antenna System with network coding leads to better diversity performance, at a lower hardware cost and higher spectral efficiency. Lihua et al. in [41] gave zero-error network coding theorems for acyclic communication networks. More specifically, they studied the problem of under what conditions a set of mutually independent information sources can be faithfully transmitted through a communication network, for which the connectivity among the nodes and the multicast

requirements of the source information are arbitrary, except that the connectivity does not form directed cycles. They obtained inner and outer bounds on the zero-error admissible coding rate region.

The Network Coding Home Page [42] provides a bibliography for the topics not covered in this thesis. Ho's dissertation [34] is also an excellent reference for network coding.

Chapter 4

Network switching

This chapter gives a detailed description of the field of network switching as well as summarizes pioneering work done in this field. Network switching is a special case of network coding wherein, a node having a capacity of say k units, is given data more than k units at a time to transmit. In this case, the node has to switch between the incoming data streams each time. One of the basic problem that arises out of it is that what can be the optimum switching strategy for a single source multi sink multi-cast network with each edge having unit capacity.

4.1 Introduction to Network Switching

Shannon once defined communication as, "The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point." Information can be compared to fluid flow as both exhibit similar properties. However, there is a a fundamental difference between the two as shown in figures 4.1 and 4.2.

These figures clearly show that the information flow like fluid flow is not conserved and hence maximum information that could be transmitted from a source to a set of sinks cannot be

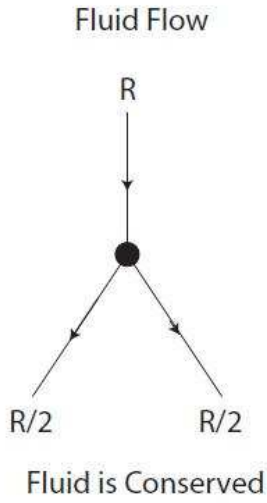

Figure 4.1: Example of Fluid Flow

given by classical max flow min cut theorem that works for fluid networks.

We explain this statement with an example. Consider figure 4.3. There is a single source and two sink nodes. Assume each edge has unit capacity and we examine the network in steady state. Now the source sends packet A on both its out going edges and in the next cycle it sends a packet B on its left edge and C on its right edge. Now each sink receive packet A from mutually edge disjoint paths. However, since the other path to each sink has a common edge, so there would be path switching and each sink would receive 1 packet in two cycles. Hence the rate due to network switching would be $\frac{3}{2}$.

4.2 Previous work

Although there is lot of work done in area of design and development of network switches and network switching in multicast networks [43], there is minimal work done in determining network switching gain for a single source multi sink communication network. We organize the work done in this specific area by the kind of approaches taken to provide solution to the problem of determining the maximum rate achieved by network switching for a single source multi sink communication network.

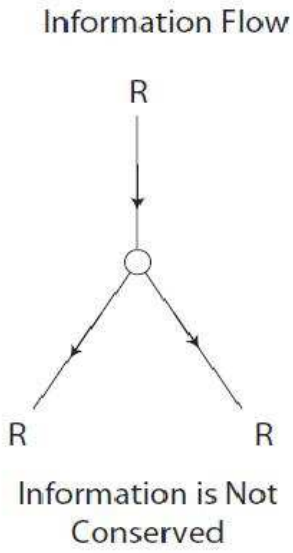

Figure 4.2: Example of Information flow

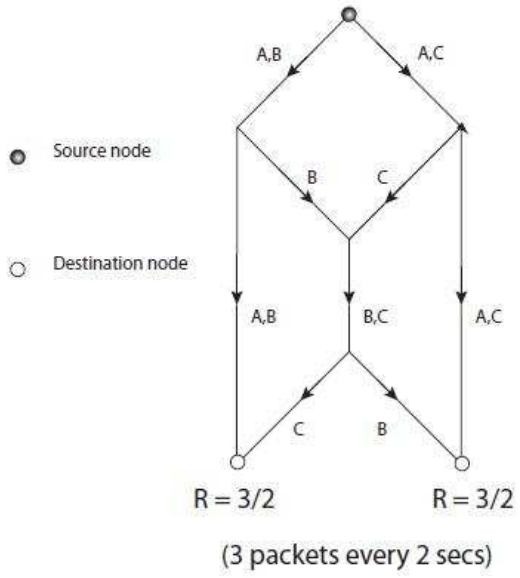

Figure 4.3: Example of Network switching

4.2.1 Information theoretic characterization of Network switching

Joy et al. in [27] gave a information theoretic definition of max-flow min-cut theorem as the rate of flow of information across any boundary is less than the mutual information between the input on one side of the boundary and output on the other side conditioned on input on the other side. Ahlswede et al. in [1] showed that it is not optimal to regard information to be multicast as a fluid that can be routed or replicated easily. They argued that it is not optimal to treat a node in a multicast network as a simple switch which could just replicate or forward the incoming flow, rather every node should act as an encoder and can perform linear functions over incoming data streams. They showed that max flow min cut theorem applies for information network after some modifications and network switching alone cannot achieve the max flow bound. In simple terms, the main result of Ahlswede et al. can be stated as follows.

Theorem 4.2.1. Let $G = (V, E)$ be a graph with source s and sinks $t_1, \ldots, t_L$ and the capacity of an edge (i, j) is R_{ij}. Let the information rate of the source (in bits per unit time) be h. Then $(\mathbf{R}, h, G)$ is admissible if and only if the values of a max-flow from s to t_l, $l = 1, \ldots, L$ are greater than or equal to h, the rate of information source.

The proof for this requires construction of a special class of block codes call α codes and can be found in [1]. However to graphically illustrate it, consider figure 4.4 which shows a single source three sink network. So here $L = 3$. The value for max flow from source to all sinks for this network is 2. Now if we use plain network switching and no coding then, we cannot send more than one common packet to all 3 sinks. Figure 4.6 shows this case. To enable all sinks to receive both the packets, s has to send b_1 and b_2 to node C in two separate cycles. However, if we deploy simple XOR based coding by sending a combination $(b_1 \bigoplus b_2)$ on node C, we can achieve a rate of 2 packets per cycle. This confirms the max-flow min cut theorem given by the author.

Based upon Ahlswede et al. work [1], if we consider network switching as a special form of network coding, then a unique problem related to performance of switching against coding can be framed. Under what conditions can network switching achieve a rate equal to network

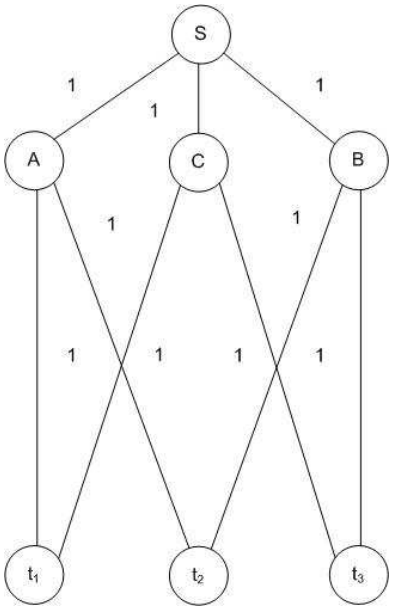

Figure 4.4: A one source three sink network

coding and what is the value of switching gap or coding gain [6] (defined as the ratio of information rate achieved by network coding and information rate achieved by network switching) for a given single source multi sink multicast network. Ngai et al. in [6] have constructed a special type of networks (refer figure 4.4) which they termed as combination networks and then determined the switching gain for them. They concluded that the network coding gain for generalized combination network can be unbounded. However it is specific to only a particular type of network and did address the problem of finding network switching rate for a generic network.

Xue-Bin Liang [7] made some progress in this problem by finding the switching gap for classical butterfly network as given in [1] in terms of capacity of the edges of the graph modeling the multicast network. The author used the fundamental theorem on max flow min cut for information networks [1] to determine the maximum information rate given by network coding. It enumerated all minimum cuts for both source sink pairs $s - t_1$ and $s - t_2$ and equated the rate due to network coding R^{**} as minimum of these value. To determine, the rate achieved by network switching, R^*, for this network, they used the results in [8] and equated R^* as the reciprocal of the value of the game with payoff matrix formed by taking

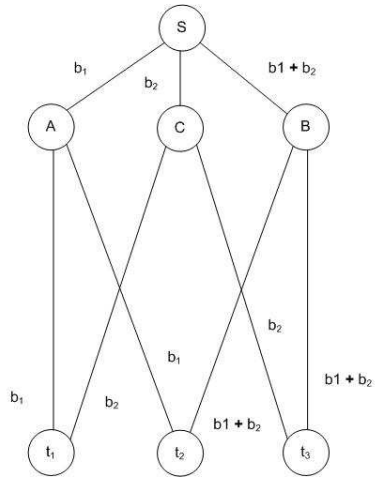

Figure 4.5: A one source three sink network with coding

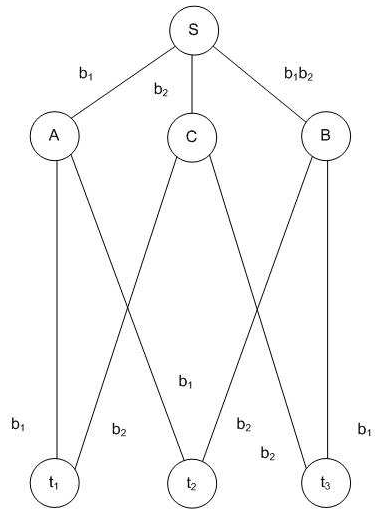

Figure 4.6: A one source three sink network with switching

reciprocal of the capacities of edges of the graph formed by the network. Liang in [7] also found the conditions on link capacities for which the switching gap becomes 1. However, he concluded that it is hard to calculate switching gap for a generic single source multicast network with arbitrarily edge capacities and it is still an open problem.

4.2.2 Game Theoretic characterization of Network Switching

From Liang's work [7], network switching is formulated as a special game called matrix game, which is a 2 person zero-sum game played by two players, such that to send flow, the first player aims at choosing an edge which has maximum capacity and the second player aims at choosing a route from source to all sinks (essentially a multicast tree) which has overall minimum weight. Both Liang [8] and R. J. McEliece in [13] have given a game theoretic characterization of network switching at nearly the same time.

R. J. McEliece in [13] gave a probabilistic solution to the problem of single source multicast problem which was based on game theory concepts. The multicast network considered is single source multi sink network depicted as a acyclic directed graph $G(V, E, c)$ where c is the capacity function of the edges. The basic underlying assumptions are that the network contains no cycles, all links operate asynchronously with noiseless transmission. He first constructed all possible spanning tree for the given network and then introduced the notion of minimum cut in the network by assigning a probabilistic value to each of the spanning tree in the network. He assigned weights to edges of the trees such that sum of weights of all edges in every multicast tree is greater than or equal to 1. He also introduced the concept of spanning tree packing function, $w(T)$ such that, for every edge belonging to the graph G i.e. $e \in G$, sum of trees having that e should be less than or equal to 1. These two are contrasting aims and hence the author used game theory principle to construct a 2 person zero sum game and the value of the payoff of this game is the reciprocal of network switching rate of the underlying network. However, the whole method is probabilistic in nature and cannot be applied in generic networks. More over assigning weights to each edge of the graph is very hard to perform.

Xue-Bin Liang in their classical paper [8], has developed an analytic framework for treatment of the achievable information rate region for single and multisource network switching for the multicast networks with links having arbitrary positive real-valued capacity and formulated the multisource network switching as a matrix game. He obtained a game-theoretic characterization of the maximum achievable information rate region for such a network and then goes on to calculate the maximum achievable information rate of network switching for butterfly network given in [1]. However, this method is again based on game theory principles and lacked a formal mathematical and graph theoretic solution. Further both these works did not addressed the problem of giving an optimal switching strategy for a generic single source multi cast network. In the next section, we explored the graph theoretic solution to this problem.

4.2.3 Fractional Steiner tree packing problem

Xue-Bin Liang in [8] stated that the problem of finding the maximum achievable information rate for single-source network switching is equivalent to the maximum fractional Steiner tree packing problem investigated by Jain, Mahdian, and Salavatipour in [28] and Wu, Chou, and Jain in [44]. Kamal Jain et al. in [28] have defined the steiner packing problem as finding the maximum number of edge-disjoint subgraphs of a given graph G that connect a given set of required points S. Single source multisink network switching problem is equivalent to steiner tree packing problem which can be defined as finding set of maximal size edge-disjoint steiner trees connecting all the sinks with the source. If the graph have real valued capacity function then it becomes fractional steiner tree packing problem. Formally, the maximum number of edge disjoint steiner trees for a Graph $G(V, E)$ is equal to the integer part of the minimum value of $\frac{E_G(P)}{\|P\|-1}$ for every (non empty) partition P of V such that $P = \{V_1, ..., V_t\}$ and $E_G(P)$ denotes the number of edges between distinct classes of P and $\|P\|$ denotes number of classes of P. The problem of fractional steiner tree packing problem is same as the problem of finding maximum switching rate for a single source multicast network having real values capacity function. The authors have formulated it as a linear program.

Lets say, τ denotes the collection of all steiner trees in graph G connecting set S, and c_e is the capacity of the edge e, then fractional steiner tree packing problem is

$$\begin{aligned} \text{maximize} \quad & \sum_{T \in \tau} x_T \\ \text{subject to } \forall e \in E \ : \ & \sum_{T: e \in T} x_T \leq c_e \\ \forall T \in \tau \ : \ & x_T \geq 0 \end{aligned} \tag{4.1}$$

Using principle of duality, 4.1 is equivalent to

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e y_e \\ \text{subject to } \forall T \in \tau \ : \ & \sum_{T: e \in T} y_e \geq 1 \\ \forall e \in E \ : \ & y_e \geq 0 \end{aligned} \tag{4.2}$$

This problem is nothing but assignment of non negative weights to graph edges such that minimum weight steiner tree has atleast unit weight and a linear (simple product) function of edges is minimized. This LP is not solvable in polynomial time because the separation oracle for this is the steiner tree problem itself. They have given an approximation algorithm for this problem using separation oracle from known results.

The research focus then shifted to finding the coding advantage for single source multicast network when coding is employed to the case when normal switching is used. Agarwal and Charikar in [45] found a remarkable parallel between the coding advantage network examples in network coding and the integrality gap examples in optimization for computer science and demonstrated that the maximum switching gap for a single-source multicast network equals the maximum integrality gap of linear programming formulation for a Steiner tree packing problem. They analyzed for both directed as well as undirected networks and helped in improving the known bounds on coding advantage for single source multicast networks.

Since the minimum weight Steiner tree problem is NP-hard for both undirected and directed networks, polynomial time solvable LP relaxations of Steiner tree are commonly used to obtain a lower bound on the optimal Steiner tree weight. The quality of the bound provided

by the LP relaxation is measured by its integrality gap, i.e. the ratio between the optimum Steiner tree weight and the optimum solution to the LP relaxation. Agarwal et al. in this work have shown that the maximum coding advantage is equal to the integrality gap of the bidirected cut relaxation for the undirected Steiner tree problem and for directed networks, the coding advantage is equal to the integrality gap of a natural LP formulation of directed Steiner tree.

To understand this work, we redefine the network as given in [45]. The input network is represented as an undirected graph $G = (V, E)$. Let $c : E \rightarrow R^+$ be capacity assignment to the edges. Let c_e denote the capacity of edge $e \in E$. Let source be m_0 and receivers $\{m_1, \ldots, m_k\}$. Let rate of network switching be denoted by $\Pi(G, c)$ and of that of network coding be denoted by $\chi(G, c)$. Let τ be the set of steiner trees for G connecting terminals $\{m_0, \ldots, m_k\}$. Coding advantage is defined as ratio $\frac{\chi(G,c)}{\Pi(G,c)}$. The Steiner packing number is given by the following linear program.

$$\begin{aligned} \text{maximize} \quad & \sum_{t \in \tau} x_t \\ \text{subject to } \forall e \in E \ : \ & \sum_{t \in \tau : e \in t} x_t \leq c_e \\ \forall t \in \tau \ : \ & x_t \geq 0 \end{aligned} \tag{4.3}$$

We can construct the dual of 4.3 as follows.

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e y_e \\ \text{subject to } \forall t \in \tau \ : \ & \sum_{t : e \in t} y_e \geq 1 \\ \forall e \in E \ : \ & y_e \geq 0 \end{aligned} \tag{4.4}$$

Let w_e to denote the weight of edge $e \in E$. For each undirected edge $e \in E$, lets have two directed edges e_1 and e_2 which represent the both orientations of e and have same weight as the undirected edge and Cost is $C(e) = C(e_1) + C(e_2)$ (refer fig 4.7). Let $D = \{e_1, e_2, \forall e \in E\}$. After orientation of the entire edge set E, E becomes a directed link set D, with the number of links in the set doubled. Since the capacity of a directed link must be non-negative, we also have $C(r) \geq 0$, $\forall r \in D$.

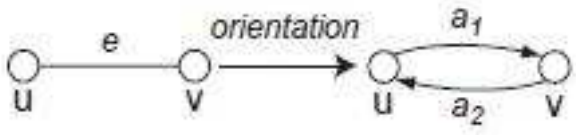

Figure 4.7: Orientation of an undirected edge in the network into two directed ones [45].

They have converted this standard steiner tree LP in terms of maximum flow in the network [65]. Let m_0 be the source and $\{m_1, \ldots m_k\}$ be the multicast receivers. $f^1, \ldots, f^k$ be the information flows to each of the receivers. Let each f^i specifies a flow rate $f^i(e)$ for each directed link $e \in D$. Let $f^i_{in}(v)$ denotes the total incoming f^i flow rate at a node v, and $f^i_{out}(v)$ be total outgoing flow at v. We need to maximize overall all flow rate f^*. We use the following flow constraints to define the flow based LP.

- Flow rates must be non-negative and upper bounded by link capacities.
- Total incoming flow f^i equals to outgoing flow rate in f^i.
- The incoming flow rate at the source and the outgoing flow rates at the receiver are all zero, for each f^i.

The last constraint is that conceptually the flow at every sink node should be the same.

$$
\begin{aligned}
& & \text{maximize} \quad & f^* \\
& \text{subject to} \quad \forall a \in D \quad & c_a \quad &\geq \quad 0 \\
& \forall e \in E \quad & c_{e_1} + c_{e_1} \quad &= \quad c_e \\
\forall i \in [1, \ldots, k], \forall v \in V - \{m_o, m_i\} \quad & & f^i_{in}(v) \quad &= \quad f^i_{out}(v) \\
& \forall i \in [1, \ldots, k] \quad & f^i_{in}(m_o) \quad &= \quad 0 \\
& \forall i \in [1, \ldots, k] \quad & f^i_{out}(m_i) \quad &= \quad 0 \\
& \forall i \in [1, \ldots, k] \quad & f^* \quad &= \quad f^i_{in}(m_i)
\end{aligned}
\tag{4.5}
$$

The solution for 4.5 i.e. f^* would be $\chi(G, c)$. Now, let $OPT(G, w)$ denotes the weight of the minimum weight Steiner tree on G for a given edge weight function w. Finding $OPT(G, w)$

is NP-hard [28]. So far, we have defined the LP on original edge set E of the graph. Now, we have a bidirectional edge set D after orientation of E, so we define bidirected cut integer program to find $OPT(G, w)$. Set C is a subset of vertices such that C contains the source m_0 and $\bar{C}$ contains at least one terminal. $\delta(C) = \{(u, v) \in D : u \in C, v \notin C\}$. The following integer program computes the minimum weight Steiner tree:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in D} w_e c_e \\ \text{subject to } \forall \text{ valid sets C} \quad : \quad & \sum_{e \in \delta(C)} c_e \geq 1 \\ \forall e \in D \quad : \quad & c_e \in \{0, 1\} \end{aligned} \tag{4.6}$$

Now solution for this LP will be $OPT(G, W)$. If we replace the last constaint in 4.6 by this

$$\forall e \in D \qquad c_e \geq 0.$$

After this relation, let the optimum value of the bidirected cut relaxation for G with given weights w be denoted by $B(G, w)$. The maximum value of the ratio $OPT(G, w)/B(G, w)$ is called the integrality gap, g of the LP relaxation.

We know give the main result of this paper.

Lemma 4.2.2. $\sum_{e \in E} c_e y_e$ is an upper bound on the value of the bidirected cut relaxation for the Steiner tree instance with edge costs given by the y_e's.

Proof. Since the graph G with capacities c_e has a flow value, $f^* = 1$, every edge can be bidirected and the capacity distributed between the two oppositely directed copies so that a flow of 1 can be routed separately from the source to every terminal according to these capacities. Further the directed capacity across any cut separating the source and at least one terminal node must be at least 1 for f^* to be 1. □

Theorem 4.2.3.

$$\max_c \frac{\chi(G, c)}{\Pi(G, c)} \leq \max_w \frac{OPT(G, w)}{B(G, w)}$$

Proof. Lets consider the equation 4.5. We can scale the capacities of the edges to make f^* as 1. Now, consider the dual to the Steiner packing LP in 4.4. By strong duality, the optimum value is equal to $\Pi(G,c)$. Consider the parameter y_e as edge cost in 4.4. The first constraint of the dual gives us the condition that every Steiner tree under these edge costs should have cost at least 1. So, using lemma 4.2.2, there exists a weight function w such that $OPT(G,w) \geq 1$ and $B(G,w) \leq \Pi(G,c)$. □

Lemma 4.2.4. $\chi(G,c)$ for the graph G with capacities c_e is atleast 1.

Proof. The bidirected cut relaxation equation 4.6 shows us a way to distribute c_e amongst forward and back edge for original edge e, such that for every cut separating the source and a terminal, the directed capacity is atleast 1. This means that the directed graph can support a flow of at least 1 from the source to every terminal. □

Theorem 4.2.5.

$$\max_c \frac{\chi(G,c)}{\Pi(G,c)} \geq \max_w \frac{OPT(G,w)}{B(G,w)}$$

Proof. Consider a network and its graph G with weight function w for which the bidirected cut relaxation has gap g. Using the result in lemma 4.2.4, we now construct a graph which coding advantage atleast g.

After orientation step, we have obtained two edges e_1 and e_2 for every edge e in original graph G such that $c_e = c_{e_1} + c_{e_2}$. Run the flow LP as given in equation 4.5 on G with these capacities and it will have coding advantage atlesast g. □

Using the results 4.2.3 and 4.2.5, we can say that for a given weight function w, there is a steiner tree of cost at most $\frac{\sum w_e c_e}{\Pi(G,c)}$. But the solution to LP (4.6) with modified constraint was $B(G,c)$ which is same as $\sum w_e c_e$. And for a given weight function w, the integral gap is originally defined as $\frac{OPT(G,c)}{B(G,c)}$. This implies that there is a Steiner tree of cost at most $\frac{B(G,c)}{\Pi(G,c)}$. This inturn implies, $\Pi(G,c) \leq 1/g$, which makes coding advantage $\frac{\chi(G,c)}{\Pi(G,c)} \geq g$.

4.2.4 Heuristic based approach for network switching problem

Single source multicast routing refers to the transmission of a packet from a single source to more than one destinations such that only one copy of the packet flows on a single link. In single source multicast routing problem, there is one source multicast node and the remaining are destination nodes. The paths between the source to destinations are represented in terms of a Multicast tree [47]. The problem of finding multicast tree is represented in terms of an undirected graph in which minimum cost paths between a source to destination multicast nodes are desired. Several approaches are being used to solve this problem and among them most popular is Steiner tree problem. In Steiner tree problem a multicast tree is found in which paths from the source to destination multicast nodes are found separately. The Steiner tree problem is NP-complete [28] and heuristic approaches are used to find solutions. Instead of heuristic approach, Umair et al. in [46] presents an algorithm to find single source multicast trees that have an over all near minimum cost paths from the source to destination nodes under the assumption that a minimum cost path from source to any destination node lies within k-hops. The algorithm is based on message transmission from destinations to source to find the tree. To explain the algorithm briefly, we first explain their model. The network is represented by a graph $G(V, E)$, where V is the set of nodes, and E is the set of links. Each link e has J non negative weights $wj(e)$, $j = 1$ to J and a cost $c(e)$. A path is defined as a sequence of nodes $p = (s, v_1, v_2, \ldots, v_k, d)$, where s is the source multicast node and d is any destination multicast node and remaining are intermediate nodes on the path between s and d. The weight of the path is the summation of all weights present at each link or it could be the minimum or maximum value incurred in the path from s to d. Similarly, the cost of the path is the summation of all links costs. The model is shown in Fig. 4.8, in which shaded circle nodes are multicast destinations and square node A is the source multicast node. The nodes can act either a router or a host. The packet format from destination to source is shown in 4.9.

The algorithm starts with A configured to accept packets from destination nodes. The field *Node Identity* (refer figure 4.9) contains identity of the destination multicast node that originated the packet, the second field contain the node identities of all nodes that occur in

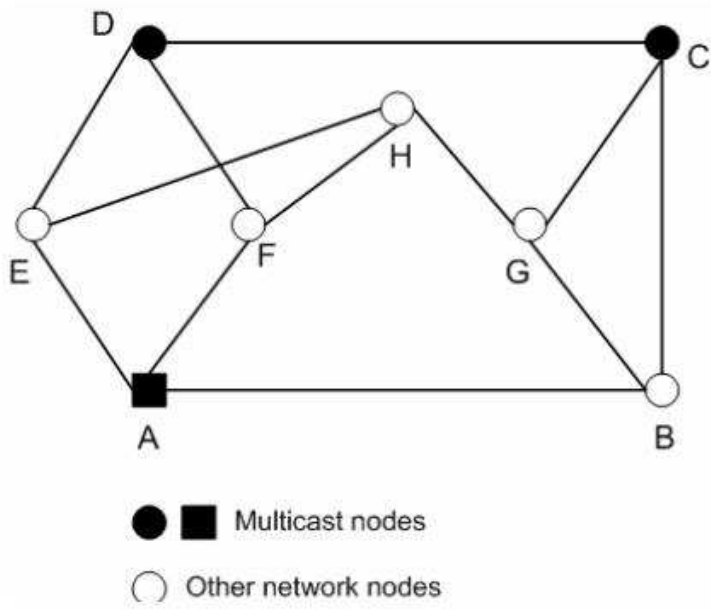

Figure 4.8: Network model for multicast routing problem [46]

Node Identity	Identities of all nodes in the path	Cost[u]= C(u,v)+Cost[u]

Figure 4.9: Packet format of destination to source [46]

the path from destination multicast node to source multicast node and the third field Cost$[u]$ contain accumulated costs of all links from destination multicast node to source multicast node where $C(u,v)$ is the cost of one link (u,v). It can also store J weights $(W1[u],\ldots,WJ[u])$ that are associated with each link. In next step, each destination sends a packet, as shown in figure 4.9, to A. The packet travels in the network until hop count reaches greater than k. In that case, it is terminated by an end host.

At router the packets copies the router identity and multiple copies of the packet are generated that are forwarded to each link. Each packet accumulates or compares with the value of the link where the router is forwarding the packet. The host in addition also checks the hop count of the packet to see if reaches greater than k and then forward the packet to all outgoing links. The packet appearing twice is discarded. Figure 4.10 shows structure of source buffer. When a packet arrives at the source multicast node, it first stores it in a

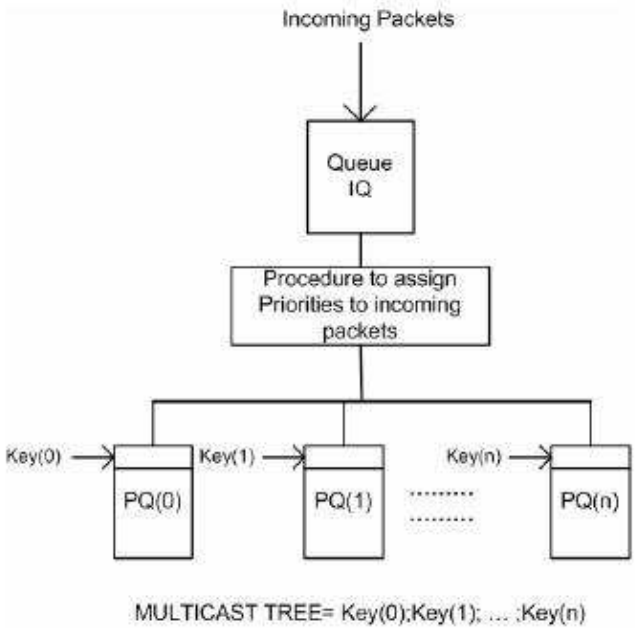

Figure 4.10: Queue structure at source [46]

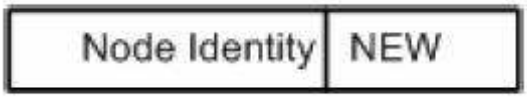

Figure 4.11: Packet format from a node to source [46]

Queue IQ. The packets are sequentially fetched from the queue and a priority is assign to the packet. When a new node wants to join the network, it will send the packet as shown in figure 4.11. The multicast tree found by the source node is shown through the $Key(0)$ to $Key(n)$ pointers when n paths to destination nodes are present.

The previous work focused on finding multicast trees used two approaches i.e. '*Dijkstra's*' algorithm and its variants, and Evolutionary algorithms. The basic approach in both the algorithms is worth describing. This algorithm was conceived by Dutch computer scientist Edsger Dijkstra and it is a graph search greedy algorithm that solves the single-source shortest path problem for a graph with nonnegative edge path costs, producing a shortest path tree. This algorithm runs in $O(|E| \lg |V|)$ time. The algorithm can be found at [64]. On the other hand, evolutionary algorithms are stochastic search methods that mimic the

metaphor of natural biological evolution. Evolutionary algorithms operate on a population of potential solutions applying the principle of survival of the fittest to produce better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. There are various forms of evolutionary algorithms i.e. genetic algorithms, genetic programming, evolution strategy etc.

The '*Dijkstra's*' algorithm has the advantage of being fast therefore it is integrated into heuristics to give a better solution in [48]. The authors [48] focused on group multicast routing problem (GMRP) which is a generalization of single-source multicasting where each member of a multicast group sends data to all other members of that group. So it is nothing but finding a set of routing trees for one tree for each member so that it can multicast messages to all members. This holds practical importance since applications like video conferences are based on it. However in this problem, two things need to be optimized i.e. tight delay constraints and sufficient bandwidth. Two approaches are suggested for this problem. The first approach is constructing a core based tree [60] which is used alone for multicasting. This tree would be rooted at the the core and having nodes as multicasting group members with minimal cost. This method suffered from the problem of too much congestion at certain links and very high upper bound on delay which make it unsuitable for video applications. The authors have instead created one source based steiner tree for each source separately so that each group member can send messages via its own tree. They have proved that this problem of creating group multicast trees for high bandwidth delay sensitive applications in point to point communication network is NP-complete and then have developed heuristics to solve it. They proposed an iterative algorithm that takes a set of multicast trees, which satisfies delay constraints but lack sufficient bandwidth, as input. It is then iteratively modified to satisfy bandwidth sufficiency condition. However their algorithm do not always give a feasible solution satisfying all QoS constraints and hence there is a need for QoS sensitive algorithm for this problem.

It is known that the '*Dijkstra's*' algorithm finds network path starting from source node

and follows links having minimum cost at each link which sometimes lead to high execution time and does not always result into a minimum overall cost path found between source to destination because of its greedy property. To overcome this problem of the algorithm, Geng Fe [50] proposed using it in forward, reverse and lookahead directions and given a QoS sensitive algorithm which gives a feasible solution with very probability. Aissa and Mnaouer [51] proposed an extended prime '*Dijkstra's*' algorithm (EPDT) that has cost and radius as two different parameters and attempt to find optimized paths with respect to both these parameters.

Moving on to the work done in evolutionary schemes, Chun-Wei Tsai et al. in [49] proposed a novel genetic algorithm for this problem. Genetic algorithm works by searching a solution from a large sample set and and then optimizing it. The authors in [49] used three generic concepts of genetic algorithms to design their scheme which are as follows.

1. Multiple search iterations in different directions to avoid local minima.

2. Different search methods for different populations in search space.

3. Use message passing to search in large solution space and upgrade the sub optimal solution to more optimal one.

To explain it further, we would formally represent the problem as well as explain the connection between the multicast routes and the chromosomes used in genetics.

Problem stmt: Given a diag $G(V, E, w, io)$ with V as set of vertices and E as set of edges between the connecting pair of vertices. Let w be the cost function $w : E \rightarrow R$. In addition, a non empty set $N = \{v_o, u_1, u_2, u_3, \ldots, u_k\}$ of terminals in G is given where $N \subseteq V$, v_0 is the source node, and $D = \{u_1, u_2, u_3, \ldots, u_k\}$ is the set of destination nodes. The delay of each vertex(io) is the sum of its reception data and transmission data i.e. $io = sn + dn$.

Objective: A subnetwork $T_G(N) = (V_T, E_T, w_T, io_T)$ of G such that : $N \subseteq V$; there is a

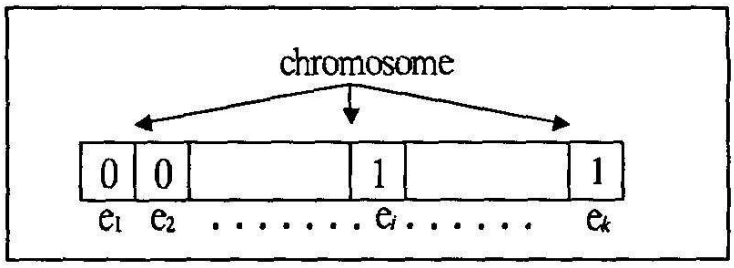

Figure 4.12: Representation of multicast tree as binary string [49]

path from source node to each destination node; the cost of $T_G(N)$, $\sum_{e_l \in E_T} w(e_l) + \sum w(i0)$, is minimized. The fitness function is $F(a) = \min \{ \sum_{e_l \in E_T} w(e_l) + \sum w(i0) \}$.

A route is defined here as feasible connection between a source and destination pair. The algorithm takes as input a routing table having R routes for each source destination pair. For a given source node v_o and destination nodes set $D = \{u_1, u_2, u_3, \ldots, u_k\}$, a chromosome can be denoted by a string of letters of length k. A gene, d_i, which is the smallest unit of a chromosome, is defined as an integer in $\{0, 1, \ldots, (R-1)\}$ and it denotes route between source and destination. The authors have numbered the edges of a graph G from 1 to k and then any multicast tree T is defined as binary string $(e_1, e_2, \ldots, e_k)$ such that $e_i = 1$ iff edge $i \in T$ and 0 otherwise. Figure 4.13 shows relationship between chromosome, gene and routing table and figure 4.12 shows a representation of multicast tree as a binary string.

A chromosome intuitively represents a multicast solution since it guarantees a path from source to a destination. In case there is a cycle then by definition of fitness function, it will have a large fitness value and would be removed from search space. The algorithm is to create more and more chromosomes satisfying fitness criteria which is nothing but a modified representation of Qos criteria.

The work proposed by LIU Ying and WU Jianping [53] is also based on evolutionary algorithms but they modified the problem from [49] by adding degree constraint to the amount of information a node (switch) can multicast steiner tree problem. The degree constraint d means that a node can output the received packet to at most $(d-1)$ out going links. The work proposes a genetic algorithm for solving degree constrained multicast routing problem.

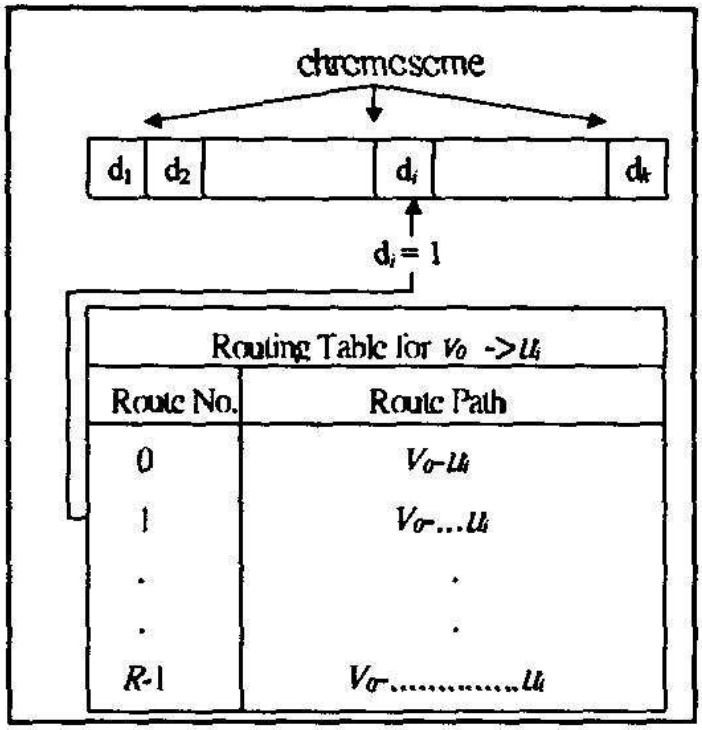

Figure 4.13: Relationship between chromosome, gene and routing table source [49]

4.2.5 Switching in multicast routing in real time networks

Switching in a network has been a core design problem in network switches. In simple terms, a network switch is a computer networking device that connects network segments. Switches use the concept of switching between different incoming streams. The problem of switching becomes all more important in dynamic multicast routing specially since multimedia applications are increasingly in demand. People are advocating for faster, incremental and dynamic schemes for constructing multicast trees. Doar and Leslie in [56] analyzed the problem of brute force multicast routing. They have talked about three ways in which heuristic Steiner trees may be used for routing multicast groups whose membership is dynamic. One is to reroute the whole tree whenever the group membership changes. A variation of this method is to permit partial or local rearrangement of the tree when modifications to the membership occur. The second approach is to begin with an optimal or near optimal tree and make minimal changes to it as group membership changes. However, it is intuitive to comment that

as membership in the group changes, one would expect the inefficiency of this method to degrade. Kou at al. in [63] have adopted this heuristic based solution to generate minimum cost steiner trees. They took a minimum spanning tree of the whole graph and modified it to remove superfluous nodes (nodes which have been removed from the network) and then tried to include more Steiner nodes, which may reduce the overall cost of the spanning tree.

The third category of multicast routing is to choose a suboptimal tree which will be resilient to changes. In the extreme case, finding the minimum source to destination path for all destinations independently and taking the union of these paths produces trees which will be resilient to changes. Doar and Leslie in [56] have examined the performance of the third method on randomly constructed graphs. They constructed multicast tree by finding the shortest distance path from source to the node which was to be added (from the add request) and add all the nodes in the path as steiner nodes to form the multicast tree. They concluded that their algorithm of computing multicast steiner trees performances worse then the standard optimal steiner trees created by Kou at al. in [63] by only a small factor.

Depending upon the nature of application and the constraints associated with it, various algorithms have been proposed to construct multicast trees for a given communication network. For applications where end to end delay is most important factor such as stock market network, shortest path routing strategies perform well. A shortest path routing strategy connects the source of the multicast to each receiver using the shortest unicast route from source to the receiver in the underlying network be it IP, ATM or ethernet. The three commonly used multicast routing protocols for IP based systems are DVMRP [57], CBT [60] and PIM [59]. For applications that require large amount of bandwidth but can tolerate the delay, the dynamic greedy algorithm as given by Imase and Waxman in [58] works very well. It gives multicast trees with good total cost. Imase and Waxman also proved in [58] that no other dynamic algorithm can perform more cost effective trees than their scheme, if the condition that multicast receivers are allowed to join the multicast groups at arbitrary times and not allowed to leave. All these results helped researchers conclude that even in the worst case, the tree produced by shortest path strategies can be up to a factor of k worst than the tree produced by best available greedy scheme.

However, greedy schemes do not find wide application. One of the reasons is that they fail to work in dynamic network conditions when nodes are free to join and leave the network at any time [58]. In this paper, the authors have studied the problem of routing and multicast communication in a wireless network in which nodes enter and leave the network in adhoc manner. They studied both the cases when existing routes are rearranged to maintain connectivity and when the rearrangement is not allowed. For the non rearrangeable version, they developed a polynomial time algorithm with worst-case performance within two times optimal of any non rearrangeable algorithm. For the rearrangeable problem they developed a polynomial time algorithm whose performance is within eight times optimal. This result showed that greedy scheme do not work well in dynamic heterogenous network systems.

Another reason of non applicability of greedy schemes is that such schemes do not guarantee an end to end delay bound. Thus such schemes are not used for high end video based applications which have high bandwidth requirements as well as stringent end to end delay requirements. On the other hand shortest path based strategies offer definite end to end delay bound. Third reason for non applicability of greedy schemes is that underlying IP network makes implementation of shortest path algorithms easier. For instance, MBone is multicast backbone which helps transmit packet to all receivers simultaneously in Internet [61, 62]. It does not support source routing and require shortest path routing. In IP Multicast, packets are forwarded along a distribution tree, rooted at the source of the data and extending to each receiver in the multicast group. Senders need not know explicitly about receivers and receivers need not know about senders. Instead, a sender simply transmits packets to an IP group address and receivers tell the network (via the Internet Group Management Protocol or IGMP) that they are interested in receiving packets sent to that group. Moreover, the process by which receivers join and leave multicast groups is timely and efficient. In Internet communication, because the source address uniquely identifies the spanning tree, multicast routers can use it to compute the forwarding decision for a given packet i.e. to forward a packet, a multicast router indexes a routing table using the packets source address, which yields a routing entry containing a set of outgoing links. The router then transmits a copy of the packet along each outgoing link. To avoid sending traffic where there are no interested

receivers, a multicast router omits links that have no downstream receivers for a particular group. In short, the source address determines the routing decision and the destination address determines the prune decision. However in MBone framework, shortest path routing is used. An IP router has two or more interfaces that attach it to multiple networks. Because of these multiple attachments, a router can switch or forward packets between networks. Each router maintains a table of routes that maps a destination network to an outgoing interface. When a packet arrives on an incoming link, the router locates the proper route and forwards the packet toward its destination on the corresponding interface. Routers exchange control messages to effect a distributed algorithm that causes each router to learn its local representation of a global set of paths through the network. Typically, this routing protocol is essentially a *shortest path* which causes the aggregate set of routing tables to converge to a state where any packet will traverse between any two hosts attached to the network.

Chapter 5

Switching gap for special class of butterfly networks

This chapter gives the analysis of ashlewede's butterfly network [1]. We have constructed different variations of classical butterfly network and calculated the switching gap for each of them. Based upon the results, we have made an intuitive observation about generic version of this butterfly network and given as a conjecture. However to understand the analytical treatment, we need to revisit the max flow min cut theorem as well as game theory.

This whole problem is a variant of classical max flow problem in graph theory applied to information theory. So, we begin this chapter with a brief discussion on max-flow min-cut theorem.

5.1 Max-Flow Min-Cut Theorem

In graph theory, a network flow is an assignment of flow to the edges of a directed graph, called a flow network in this case, where each edge has capacity (which may be positive real or integer), such that the amount of flow along an edge does not exceed its capacity. Further, there is restriction that the amount of flow into a node equals the amount of flow out of it,

except if it is a source, which only has outgoing flow, or sink, which has only incoming flow. A flow network can be used to simulate traffic in a road system, fluids in pipes, currents in an electrical circuit, or anything similar in which something travels through a network of nodes.

Given a graph G (V,E) with nodes V and edges E, and special nodes source s (in-degree 0) and sink t (out-degree 0). Let $f(u,v)$ be the flow from node u to node v, and $c(u,v)$ the capacity. Formally stating, a network flow is a real function f: $V \times V \rightarrow R$ with the following three properties for all nodes u and v:

1. Skew Symmetry: $f(u,v) = -f(v,u)$

2. Capacity Constraints: $f(u,v) \leq c(u,v)$

3. Flow conservation: $\sum_{w \in V} f(u,w) = 0$, where $w \notin (s, t)$

The residual capacity of an edge is $c_f(u,v) = c(u,v) - f(u,v)$. A residual network is denoted by $G_f\,(V, E_f)$ and it consists of residual edges obtained by transforming the original network G. This way there can be an edge from v to u in the residual network, even though there is no edge from u to v in the original network.

A cut (S,T) of flow network $G=(V, E)$ is a partition of V into S and $T = V\text{-}S$ such that $s \in S$ and $t \in T$.

An augmenting path is a path $(u_1, u_2, \ldots, u_k)$, where $u_1 = s$, $u_k = t$, and $c_f\,(u_i, u_i + 1) > 0$, such that more flow could be pushed along this path. There are various ways of choosing an augmenting path and depending upon that different max flow determination algorithms have been proposed.

5.1.1 The Maximum-Flow Min-Cut Theorem

Theorem 5.1.1. (The Max-Flow Min-Cut Theorem) If f is a flow in a flow network $G = (V, E)$ with source s and sink t, then the 3 following statements are equivalent

1. f is a maximum flow in G.
2. The residual network G_f contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.

5.2 Game Theory

Game theory is a branch of mathematical analysis developed to study decision making in conflict situations. Such a situation exists when two or more decision makers who have different objectives, act on the same system or share the same resources. There are two person and multi person games. Game theory provides a mathematical process for selecting an *Optimum strategy* (that is, an optimum decision or a sequence of decisions) in the face of an opponent who has a strategy of his own. In game theory one usually makes the following assumptions:

1. Each decision maker *player* has available to him two or more well-specified choices or sequences of choices called *plays*.
2. Every possible combination of plays available to the players leads to a well-defined end-state (win, loss, or draw) that terminates the game.
3. A specified payoff for each player is associated with each end-state (a *zero-sum* game means that the sum of payoffs to all players is zero in each end-state).
4. Each decision maker has perfect knowledge of the game and of his opposition; that is, he knows in full detail the rules of the game as well as the payoffs of all other players.

5. All decision makers are rational; that is, each player, given two alternatives, will select the one that yields him the greater payoff.

Game Theory finds wide application in areas of financial accounting, economics, sociology, computer networks, stochastic based applications etc. Network switching can be formulated as a special game, called matrix game played in the multicast networks [8, 13]. Interested readers can refer [10, 11, 14] for further details.

5.2.1 Zero-Sum games

Formally, a game τ is said to be zero-sum if and only if at each terminal vertex of the game tree, the payoff function (p_1,...,p_n) satisfies

$$\sum_{i=1}^{n} p_i = 0.$$

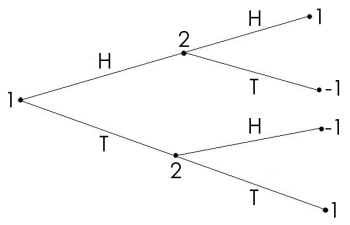

Figure 5.1: Game of matching Pennies.

Figure 5.1 shows Game Tree for matching pennies [9]. It is a simple example of zero-sum with two players. It consists of tossing a coin by each player and in case both outcomes are heads or both are tails then player I wins, otherwise player II wins.

A finite zero sum 2 person game reduces to a matrix A, with as many rows as Player PI has strategies and as many columns as player PII has strategies. In simpler terms, the payoff is defined as the amount Player PI receives from second Player PII. PI will try to maximize

it while PII will try to minimize it. It is a play of game such that, if PI chooses say, i^{th} row from the I rows and PII chooses say, j^{th} column from the J columns, then the expected payoff, is the element a_{ij}, in the i^{th} row and j^{th} column of the matrix.

We refer to rows and columns as pure strategies of PI and PII respectively. Aim of PI is to maximize the minimum payoff, thereby guaranteing a lower bound called (*gain floor*) [9] given by

$$\overline{\nu} = \max_{1\le i\le I} \min_{1\le j\le J} a_{ij}$$

Similarly, PII will try to choose a pure strategy so as to minimize the maximum payoff thereby, guaranteing an upper bound for loss (called *loss ceiling*) [9] given by

$$\underline{\nu} = \min_{1\le j\le J} \max_{1\le i\le I} a_{ij}$$

It is clear that $\underline{\nu} \ge \overline{\nu}$ and equality holds $\Leftrightarrow$ there exists a pair of strategies (i^*, j^*) satisfying the following condition.

$$\min_{1\le j\le J} a_{i^*j} = a_{i^*j^*} = \max_{1\le i\le I} a_{ij^*} \tag{5.1}$$

(i^*, j^*) satisfying (5.1) is called a *saddle* point for the payoff matrix A. It is a also termed as Nash equilibrium point of the game since, at this point both the players are having maximum gain and one way change of strategy by either player will not provide any gain. Saddle point may or may not exist. For e.g., consider this game matrix.

$$\begin{vmatrix} 5 & 1 & 3 \\ 3 & 2 & 4 \\ 3 & 0 & 1 \end{vmatrix}$$

It has a saddle point at 1^{st} row 2^{nd} column with ν=1.

5.2.2 Mixed Strategy

A mixed strategy for a player is a probability distribution on the set of his pure strategies. Precisely, if say 'm' pure strategies are there, a mixed strategy reduces to m-vector, x=(x_1, ..., x_m), satisfying [10]

$$x_i \geq 0 \quad \text{and}$$

$$\sum_{i=1}^{m} x_i = 1.$$

Let X be the set of all mixed strategies of PI and let Y be the set of all mixed strategies of PII. If PI chooses mixed strategy x and PII chooses y, then the expected payoff is

$$\begin{aligned} A(x,y) &= \sum_{i=1}^{m}\sum_{j=1}^{m} x_i a_{ij} y_i \text{ or in matrix notation} \\ A(x,y) &= xAy^T \\ \nu_{\mathrm{I}} &= \text{PI's gain floor} \\ &= \max_{x \in X} \min_{j} xA \cdot j \\ &\quad (A \cdot j \text{ is the } j^{th} \text{ column of } A) \\ \nu_{\mathrm{II}} &= \text{PII's loss ceiling} \\ &= \min_{y \in Y} \max_{i} A \cdot iy^T \end{aligned}$$

We can represent mixed strategy for PI by an I-dimensional probability distribution X = $(x_1,\ x_2,\ ...,\ x_I)^T$, where T denotes the transpose of a matrix. A mixed strategy for PII is denoted by a J-dimensional probability distribution vector Y=$(y_1,\ y_2,\ ...,\ y_J)^T$. So, the *expected payoff* in case PI chooses mixed strategy x and PII chooses mixed strategy y is

$$x^T A\, Y = \sum_{i=1}^{I}\sum_{j=1}^{J} x_i a_{ij} y_j.$$

Let us define X as

$$\begin{aligned} X &= \{(x_1, x_2, ..., x_I) \in R^I \mid \\ &\quad \sum_{i=1}^{I} x_i = 1 \text{ and } x_i \geq 0 \text{ for } i = 1, 2....I\}. \end{aligned} \tag{5.2}$$

Now we can define PI's *expected gain-floor* (ν_{I}) and PII's *expected loss ceiling* (ν_{II}).

$$\nu_{\mathrm{I}} = \max_{x \in X} \min_{y \in Y} x^T A\, Y. \tag{5.3}$$

$$\nu_{\mathrm{II}} = \min_{y \in Y} \max_{x \in X} x^T A\, Y. \tag{5.4}$$

Theorem 5.2.1. (MiniMax Theorem) It states that

$$\nu_{\mathrm{I}} = \nu_{\mathrm{II}} \tag{5.5}$$

The proof of this theorem can be found here [9]. Minimax Theorem [9] implies that every matrix game with payoff matrix as A has at least one pair of mixed strategies (x^*,y^*) where in, $x^* \in X$ and $y^* \in Y$ such that

$$\nu_{\mathrm{I}} = \min_{y \in Y} (x^*)^T A y = (x^*)^T A y^* = \max_{x \in X} x^T A y^* = \nu_{\mathrm{II}}. \tag{5.6}$$

So (x^*,y^*), is the saddle point of the expected payoff function $x^T A y$. Following Lemma [13, pg. 138, Eqn. (5)] characterizes the solution of a matrix game.

Lemma : For a matrix game with $I \times J$ payoff matrix A, a necessary and sufficient condition for a mixed-strategy pair (x^*, y^*) given $x^* \in X$ and $y^* \in Y$ to be a Nash Equilibrium point and for a real number $\nu \in R$ to be the value of the game is that every component of the vector $(x^*)^T A \in R^J$ is $\geq \nu$ and every component of the vector $Ay^* \in R^I$ is $\leq \nu$ [9].

5.2.3 Computation of Optimal Strategies

In case,if a saddle point exists, then pure strategies i and j or equivalently, the mixed strategy x and y with x_i=1, y_i=1 and all other components equal to zero, will be optimal strategies of PI and PII respectively.

Domination : In a matrix A, we say that the i^{th} row dominates the k^{th} row if

$$\begin{aligned} a_{ij} &\geq a_{kj} \text{ for every } j \text{ and} \\ a_{ij} &> a_{kj} \text{ for atleast one } j \end{aligned}$$

Similarly, we say that j_{th} column dominates the i_{th} column if,

$$\begin{aligned} a_{ij} &\leq a_{il} \text{ for every } i \text{ and} \\ a_{ij} &< a_{il} \text{ for at least one } i \end{aligned}$$

Here is a working example

$$A = \begin{vmatrix} 2 & 0 & 1 & 4 \\ 1 & 2 & 5 & 3 \\ 4 & 1 & 3 & 2 \end{vmatrix}$$

2^{nd} column dominates the 4^{th} column $\Rightarrow$ PII will never use 4^{th} column.

$$A = \begin{vmatrix} 2 & 0 & 1 & \underline{4} \\ 1 & 2 & 5 & \underline{3} \\ 4 & 1 & 3 & \underline{2} \end{vmatrix}$$

3^{rd} row dominates 1^{st} row $\Rightarrow$ PI will never use 1^{st} row

$$A = \begin{vmatrix} \underline{2} & \underline{0} & \underline{1} & \underline{4} \\ 1 & 2 & 5 & \underline{3} \\ 4 & 1 & 3 & \underline{2} \end{vmatrix}$$

3^{rd} column dominates first column. So, eliminating all dominated components, we have

$$A = \begin{vmatrix} 1 & 2 \\ 4 & 1 \end{vmatrix}$$

So we need solve just the 2×2 game-matrix.

Another way to solve for optimal strategy for matrix games is by fictious play method given in [11] but it works for integral link capacities. Moreover, it is intuitive and lacks formal basis.

5.3 Analysis for Singular Butterfly Network

Figure 5.2 shows singular butterfly network. Here $w_i > 0$ denotes the link capacities of the edges in the graph.

Now, according to Ahlswede-Cai-Li-Yeung's fundamental theorem for single-source network coding [1], the maximum achievable information rate denoted by R^{**} is equal to the minimum

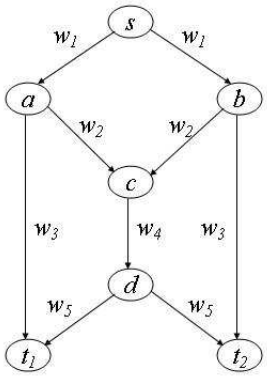

Figure 5.2: Singular-Symmetric Butterfly network.

of the s-t cuts for all source-sink pairs in the network. To enumerate the s-t cuts, consider the subgraph $G' = (V', E')$, where $V' \subset V$ and $E' \subset E$, which is formed from graph shown in above figure by removing all paths between s-t_2. All s-t cuts are shown as dashed lines in figure 5.3. The cut-set for this network is enumerated below.

1. $\{(s,a),(s,b)\} = 2w_1$
2. $\{(s,a),(b,c)\} = w_1 + w_2$
3. $\{(s,a),(a,c),(d,t_1)\} = 2w_2 + w_3$
4. $\{(s,a),(a,c),(c,d)\} = w_1 + w_2 + w_4$
5. $\{(a,t_1),(a,c),(b,c)\} = w_1 + w_2 + w_5$
6. $\{(a,t_1),(c,d)\} = w_3 + w_4$
7. $\{(a,t_1),(d,t_1)\} = w_3 + w_5$

Please note that since butterfly network is a symmetrical network so we can get the minimum cut values for $s - t_2$ from values for $s - t_1$. So, for subgraph between s and t_2, we have the following s-t cut sets.

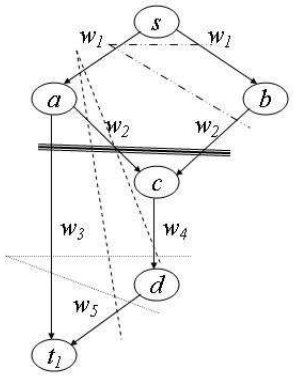

Figure 5.3: Singular-Symmetric Butterfly sub network between $s - t_1$.

1. $\{(s,a),(s,b)\}$=$2w_1$
2. $\{(s,b),(a,c)\}$=$w_1 + w_2$
3. $\{(b,t_2),(a,c),(b,c)\}$=$2w_2 + w_3$
4. $\{(s,b),(b,c),(c,d)\}$=$w_1 + w_2 + w_4$
5. $\{(s,b),(b,c),(d,t_2)\}$=$w_1 + w_2 + w_5$
6. $\{(b,t_2),(c,d)\}$=$w_3 + w_4$
7. $\{(b,t_2),(d,t_2)\}$=$w_3 + w_5$

For the given singular butterfly network, we have assumed the following conditions

$w_1 < w_2$

$w_1 < w_3$

$w_4 < w_5$

Based upon these three assumptions, we combine the various $s-t$ cut set values and finally, R^{**} is equal the minimum of the following 5 values.

1. $\min(w_1,w_3)+\min(w_4,w_5)$
2. $w_1+\min(w_1,w_2)$
3. $2w_2+w_3$
4. $w_1+w_2+w_3$
5. $w_1+w_2+w_3+w_4$

Under these conditions, we have R^{**} is minimum of

$$\begin{aligned} R^{**} &= \min(w_1 + w_4, 2w_1). \\ R^{**} &= w_1 + \min(w_1, w_4). \end{aligned} \tag{5.7}$$

Now, we are left with problem to solve R^* i.e. maximum achievable information rate by network switching (NS). Liang in [8] gives a method to compute R^* of a network by constructing its payoff-matrix and then solving it as per game theory principles [9] and finally taking reciprocal of it to get R^*. In a nutshell, first we determine multicast routes from source to each of the sink nodes. This is a path enumeration problem and can be done by constructing rooted trees for each of the link and then concatenating one path each from all such trees. Formally, a rooted tree is defined as an acyclic digraph with a unique node, called its root node, which has the property that there exists a unique path from the root node to each other node. We call the set of links, τ as a multicast route of the underlying digraph from the source node s to the sink nodes t_1, t_2, ..., t_l, if the digraph (S,τ) induced by τ is a rooted tree of G with t_1, t_2, ..., t_l, whose root node is s and whose leaves are all sink nodes. If L=1, a multicast route τ is the set of links of an open path from the source node s to the sink node t_1. Then for each multicast route τ_j they have defined an *indicator*

function over E as $\chi\tau_j(e_i)$=1 if $e_i \in \tau_j$ and 0 otherwise, for $i = 1, 2, \ldots, I$ and $j = 1, 2, \ldots, J$. They have then created an $I \times J$ payoff matrix A such that

$$a_{ij} = \frac{1}{\Theta(e_i)} \chi\tau_j(e_i)$$

for $i = 1, 2, \ldots, I$ and $j = 1, 2, \ldots, J$ with $\{e_1, e_2, \ldots, e_I\}$ being the set of links E and $\{\tau_1, \tau_2, \ldots, \tau_J\}$ being the set of multicast routes from s to $t_1, t_2, \ldots, t_l$.

Brute force method is to enumerate all multicast routes in the network, form a payoff matrix say A using all links and multicast routes enumerated above and solve it to get its value i.e., val(A). Then $R^* = 1/\text{val}(A)$. For this network given in Fig 5.1, payoff matrix is as follows.

$$A = \begin{vmatrix} 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 0 & 1/w_1 & 1/w_1 & 0 & 1/w_1 & 1/w_1 \\ 1/w_3 & 1/w_3 & 1/w_3 & 0 & 0 & 0 & 0 \\ 0 & 1/w_2 & 0 & 1/w_2 & 1/w_2 & 0 & 0 \\ 0 & 0 & 1/w_2 & 0 & 0 & 1/w_2 & 1/w_2 \\ 1/w_3 & 0 & 0 & 1/w_3 & 0 & 1/w_3 & 0 \\ 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \\ 0 & 0 & 0 & 1/w_5 & 1/w_5 & 1/w_5 & 1/w_5 \\ 0 & 1/w_5 & 1/w_5 & 0 & 1/w_5 & 0 & 1/w_5 \end{vmatrix}$$

However, this is not a feasible method for more complex networks since it will have many more links and multicast routes than the network shown in fig 5.2. We do not necessarily need an enumeration of all the multicast routes to find R^*. We need to work only on the dominating links and dominating multicast routes. A link is dominated by another link if every multicast routes including the former also includes the latter. A multicast route is dominated by another multicast route if each dominating link in the latter is also in the former. Applying successive elimination method from previous section, we can use dominated links and dominated multicast routes to get a simpler square pay off matrix, then find the value of the game and use it to get R^*.

Using assumptions about link capacities stated earlier to compute R^{**}, we have both the links with capacities w_1 and the single link with capacity w_4 as dominating and these 3 links form 3 multi cast routes. As a result, we get the following payoff matrix.

$$A_1 = \begin{vmatrix} 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 0 & 1/w_1 \\ 0 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Alternatively, using assumptions about various link capacities stated earlier, we see that row number 1 dominates row number 3 and 4 and 5; row number 1 dominates row number 5 and 6 and row number 1 dominates row number 7 and 8 and 9. Removing all dominated rows we have, the following matrix.

$$\begin{vmatrix} 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 0 & 1/w_1 & 1/w_1 & 0 & 1/w_1 & 1/w_1 \\ 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Now we see that column number 7 dominates column number 3, 4 and 6. Eliminating column number 3, 4 and 6, we have

$$\begin{vmatrix} 1/w_1 & 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 0 & 0 & 1/w_1 \\ 0 & 1/w_4 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Now, column 2 dominates column 3 or vice-versa since both are equal. Hence eliminating column number 2, we get

$$A_1 = \begin{vmatrix} 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 0 & 1/w_1 \\ 0 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Please note that the rows of this matrix shows the dominant links in the singular butterfly network where as the columns denote multicast routes. It is clear from [8] that val(A) =

val(A_1). Using Lemma given in section III and applying lagrange multipliers [12] to A, we have following cases.

1. *Case 1:* if $2w_1 \geq w_4$ then we have X^*=(w_1/(2w_1+w_4),w_1/(2w_1+w_4),w_4/(2w_1+w_4)) and Y^*=((2w_1-w_4)/(2w_1+w_4),w_4/(2w_1+w_4),w_4/(2w_1+w_4)) are optimal strategies for PI and PII and val(A) is
$$\text{val}(A) = \frac{2}{(2w_1 + w_4)}$$

2. *Case 2:* if $2w_1 < w_4$ then we have X^*=(1/2,1/2,0) and Y^*=(0,1/2,1/2) and val(A) is
$$\text{val}(A) = \frac{1}{2w_1}.$$

From these two cases, we get,

$$\text{val}(A) = \min\left((2/(2w_1 + w_4)), (1/2w_1)\right).$$

Hence
$$R^* = \frac{1}{\text{val}(A)} = \frac{2w_1 + \min(2w_1, w_4)}{2}. \tag{5.8}$$

Using this,
$$R^{**} = w_i + w_j. \tag{5.9}$$

And,
$$R^* = \frac{2}{w_i + w_j + \min\{(w_i + w_j), (w_k)\}}. \tag{5.10}$$

Using 5.7 and 5.8, we have,

$$\frac{R^{**}}{R^*} = \frac{2(w_1 + \min(w_1, w_4))}{2w_1 + \min(2w_1, w_4)}.$$

Now conditioning on values of w_1 and w_4, we have following four cases.

1. case 1: If $2w_1 < w_4$, then we have

$$\frac{R^{**}}{R^*} = \frac{4w_1}{4w_1} = 1$$

2. case 2: If $w_1 < w_4 < 2w_1$, then we have

$$\begin{aligned} \frac{R^{**}}{R^*} &= \frac{2(w_1 + w_4)}{2w_1 + w_4} \\ \frac{R^{**}}{R^*} &= \frac{2}{1 + \frac{w_4}{2w_1}} \end{aligned}$$

3. case 3: If $w_4 < w_1$, then we have

$$\begin{aligned} \frac{R^{**}}{R^*} &= \frac{2(w_1 + w_4)}{2w_1 + w_4} \\ \frac{R^{**}}{R^*} &= \frac{2}{1 + \frac{w_1}{w_1 + w_4}} \end{aligned}$$

If we have $w_1 = w_4 = w$, then the switching gap becomes,

$$\frac{R^{**}}{R^*} = \frac{2(w + w)}{2w + w} = \frac{4}{3} \tag{5.11}$$

5.4 Analysis for dual butterfly network

Figure 5.4 shows dual butterfly network. Here $w_i > 0$ denotes the link capacities of the edges in the graph.

For the above network taking into account assumptions as in case of singular butterfly network, R^{**} is equal the following 7 values.

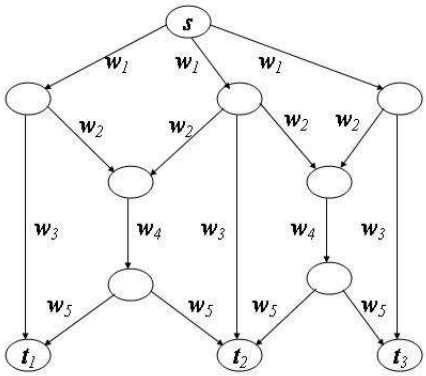

Figure 5.4: Dual-Symmetric Butterfly network

1. $3w_1$
2. w_1+w_2
3. $w_2+w_3+\min(w_1,w_2)$
4. w_1+w_4
5. $2w_4+w_3$
6. $2w_1+w_3+w_5$
7. $w_1+w_2+w_3+w_4$

From this set under the assumptions in section 5.3, we only take minimum min-cuts and thus

$$R^{**} = \min(3w_1, w_1 + w_2, w_1 + w_4). \tag{5.12}$$

Using assumptions about link capacities stated earlier to compute R^{**}, we have all three links with capacities w_1 and all two links with capacity w_4 as dominating and these 5 links form 5 multi cast routes. After elimination of dominated links and multicast routes, we get the

following payoff matrix.

$$A = \begin{vmatrix} 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 1/w_1 & 1/w_1 & 0 & 1/w_1 \\ 1/w_1 & 1/w_1 & 0 & 1/w_1 & 1/w_1 \\ 1/w_4 & 0 & 1/w_4 & 1/w_4 & 1/w_4 \\ 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Proceeding as in previous section, we use the Lemma given in section III and apply lagrange multipliers [12] to A and get following cases.

1. *Case* 1 : If $3w_1 \geq 2w_4$ and $2w_4 \geq w_1$ then X^*=(w_1/(3w_1+2w_4),w_1/(3w_1+2w_4),w_1/(3w_1+2w_4), w_4/(3w_1+2w_4),w_4/(3w_1+2w_4)) and Y^*=((3w_1-2w_4)/(3w_1+2w_4),(3w_1-2w_4)/(3w_1+2w_4), (2w_4-w_1)/(3w_1+2w_4),(2w_4-w_1)/(3w_1+2w_4), (2w_4-w_1)/(3w_1+2w_4)) are optimal strategies for PI and PII and val(A) is

$$\text{val}(A) = \frac{4}{(3w_1 + 2w_4)}.$$

2. *Case* 2 : If $3w_1 < 2w_4$ then X^*=(1/3,1/3,1/3,0,0) and Y^*=(0,0,1/3,1/3,1/3) are optimal strategies for PI and PII and val(A) is

$$\text{val}(A) = \frac{2}{3w_1}.$$

3. *Case* 3 : If $2w_4 < w_1$ then X^*=(0,0,0,1/2,1/2) and Y^*=(1/2,1/2,0,0,0) are optimal strategies for PI and PII and val(A) is

$$\text{val}(A) = \frac{1}{2w_4}.$$

From these three cases, we get

$$\text{val}(A) = \min\left((\frac{4}{3w_1 + 2w_4}), (\frac{2}{3w_1}), (\frac{1}{2w_4})\right).$$

Hence,

$$R^* = \min\left((\frac{3w_1 + 2w_4}{4}), (\frac{3w_1}{2}), (2w_4)\right). \tag{5.13}$$

Using 5.12 and 5.13, we have,

$$\frac{R^{**}}{R^*} = \frac{\min(3w_1, w_1 + w_2, w_1 + w_4)}{\min\left((\frac{3w_1+2w_4}{4}), (\frac{3w_1}{2}), (2w_4)\right)} \tag{5.14}$$

If we have w_1=w_4=w then,

$$\frac{R^{**}}{R^*} = \frac{(2w)}{(\frac{5w}{4})} = \frac{8}{5} = 1.60. \tag{5.15}$$

5.5 Analysis for triple butterfly network

Figure 5.5 shows triple butterfly network. Here $w_i > 0$ denotes the link capacities of the edges in the graph.

For the above network taking into account assumptions as in case of singular butterfly network, R^{**} is equal the following seven values.

1. $4w_1$
2. w_1+w_2
3. w_2+w_3+$\min\{w_1, w_2\}$

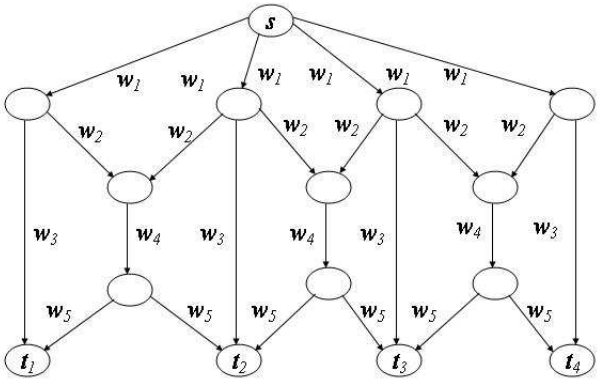

Figure 5.5: Triple-symmetric Butterfly network

4. w_1+w_4

5. $2w_4+w_3$

6. $2w_1+w_3+w_5$

7. $w_1+w_2+w_3+w_4$

From this set under the assumptions in section V, we only take minimum min-cuts and thus

$$R^{**} = \min(4w_1, w_1 + w_2, w_1 + w_4). \tag{5.16}$$

Using assumptions about link capacities stated earlier to compute R^{**}, we have all four links with capacities w_1 and all three links with capacity w_4 as dominating and these 7 links form 7 multi cast routes. After elimination of dominated links and multicast routes, we get the following payoff matrix.

$$A = \begin{vmatrix} 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 \\ 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 & 1/w_1 \\ 1/w_1 & 1/w_1 & 1/w_1 & 1/w_1 & 0 & 1/w_1 & 1/w_1 \\ 1/w_1 & 1/w_1 & 1/w_1 & 0 & 1/w_1 & 1/w_1 & 1/w_1 \\ 1/w_4 & 1/w_4 & 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \\ 1/w_4 & 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \\ 0 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 & 1/w_4 \end{vmatrix}$$

Proceeding as in previous section, we use the Lemma given in section III and apply lagrange multipliers [12] to A and get following cases.

1. *Case 1:* If $4w_1 \geq 3w_4$ and $3w_4 \geq 2w_1$ then X^*=($w_1/(4w_1+3w_4)$,$w_1/(4w_1+3w_4)$,$w_1/(4w_1+3w_4)$, $w_1/(4w_1+3w_4)$,$w_4/(4w_1+3w_4)$,$w_4/(4w_1+3w_4)$,$w_4/(4w_1+3w_4)$) and Y^*=($(4w_1$-$3w_4)/(4w_1+3w_4)$,$(4w_1$-$3w_4)/(4w_1+3w_4)$, $(4w_1$-$3w_4)/(4w_1+3w_4)$,$(3w_4$-$2w_1)/(4w_1+3w$ $(3w_4$-$2w_1)/(4w_1+3w_4)$,$(3w_4$-$2w_1)/(4w_1+3w_4)$,$(3w_4$-$2w_1)/(4w_1+3w_4)$) and val(A) is

$$\text{val}(A) = \frac{6}{4w_1 + 3w_4}.$$

2. *Case 2 :* If $4w_1 <3w_4$ then X^*=(1/3,1/3,1/3,0,0,0,0) and Y^*= (0,0,0,$(w_4$-$w_1)/w_4$,$w_1/3w_4$,$w_1/3w_4$ and val(A) is

$$\text{val}(A) = \frac{(3w_4 - w_1)}{3w_1w_4}.$$

3. *Case 3 :* If $3w_4 <2w_1$, then X^*=(0,0,0,0,1/3,1/3,1/3) and Y^* = (1/3,1/3,1/3,0,0,0,0) and val(A) is

$$\text{val}(A) = \frac{2}{3w_4}.$$

From these three cases, we get,

$$\text{val}(A) = \min\Big((\frac{6}{4w_1 + 3w_4}), (\frac{3w_4 - w_1}{3w_1w_4}), (\frac{2}{3w_4})\Big).$$

Hence

$$R^* = \min\Big((\frac{4w_1 + 3w_4}{6}), (\frac{3w_1w_4}{3w_4 - w_1}), (\frac{3w_4}{2})\Big). \tag{5.17}$$

Using 5.16 and 5.17, we have,

$$\frac{R^{**}}{R^*} = \frac{\min\left((4w_1, w_1 + w_2, w_1 + w_4)\right)}{\min\Big((\frac{4w_1+3w_4}{6}), (\frac{3w_1w_4}{3w_4-w_1}), (\frac{3w_4}{2})\Big)}. \tag{5.18}$$

If we have w_1=w_4=w then,

$$\frac{R^{**}}{R^*} = \frac{(2w)}{(\frac{7w}{6})} = \frac{12}{7} = 1.72. \tag{5.19}$$

5.6 Analysis for generic butterfly network

Generic butterfly network is the singular-symmetric butterfly network repeated n times. So it has one source and a total of $(n + 1)$ sinks. Using results from section 5.3, 5.4 and 5.5, information rate with Network Coding (NC), i.e., R^{**} will be same as that of Triple symmetric butterfly network except one term $(n + 1)w_1$ will be added in this case.

$$R^{**} = \min\{\left((n + 1)w_1, w_1 + w_2, w_1 + w_4\right)\}. \tag{5.20}$$

Using assumptions about link capacities stated earlier to compute R^{**}, we have $(n+1)$ dominant links with capacities w_1 and n dominant links with capacity w_4. Thus the payoff matrix for this network will be $(2n + 1) \times (2n + 1)$.

It is very difficult to solve this pay off matrix via game theoretic methods. Here we give an intuitive observation based on the results from previous three sections.

1. *Case* 1 : If $(n+1)w_1 \geq nw_4$ and $nw_4 \geq (n-1)w_1$ then x_1=$w_1/((n+1)w_1+nw_4)$, x_2=$w_1/((n+1)w_1+nw_4)$,...,x_n=$w_1/((n+1)w_1+nw_4)$; x_{n+1}=$w_4/(n+1)w_1+nw_4)$,..., x_{2n+1}=$w_4/($
   And y_1=$((n+1)w_1$-$nw_4)/((n+1)w_1+nw_4)$,
   y_2=$((n+1)w_1$-$nw_4)/((n+1)w_1+nw_4)$,...,
   y_n=$((n+1)w_1$-$nw_4)/((n+1)w_1+nw_4)$;
   y_{n+1}=$(nw_4$-$(n$-$1)w_1)/((n+1)w_1+nw_4)$,...,
   y_{2n+1}=$(nw_4$-$(n$-$1)w_1)/((n+1)w_1+nw_4)$.

From these conditions, it follows that

$$\text{val}(A) = \frac{2n}{((n+1)w_1 + nw_4)}$$

Hence,

$$R^* = \frac{((n+1)w_1 + nw_4)}{2n} \tag{5.21}$$

Using 5.20 and 5.21, we have,

$$\frac{R^{**}}{R^*} = \frac{\min((n+1)w_1, w_1 + w_2, w_1 + w_4)}{\frac{((n+1)w_1+nw_4)}{2n}} \tag{5.22}$$

If we have w_1=w_4=w then,

$$\frac{R^{**}}{R^*} = \frac{(4nw)}{((2n+1)w)} = \frac{4n}{2n+1}. \tag{5.23}$$

We can verify this claim by substituting n with values 1, 2 and 3 to get the coding gain for singular symmetric, dual symmetric and triple symmetric butterfly networks respectively.

Chapter 6

Conclusion and Future Work

In this work, we have studied information network flow problem on a single source multi sink multicast communication network. We have tried to extend the idea in [7] by analyzing a generic butterfly network under suitable assumptions about link capacities. From the results, we could conclude that information rate due to Network coding is coming out to be same in all versions of the network. This means as number of nodes are increasing the network, no additional gain is coming out by using Network Coding. On the other hand, information rate due to network switching is decreasing as we are duplicating singular butterfly network. Hence, simple switching for generic butterfly network does not increase throughput. Further, switching gain is equal to coding gain for the butterfly network [1] only when all link capacities are equal.

We found out that application of game theory for network switching is not generic and requires certain conditions on link capacities. Moreover the problem for single source multicast flow appears to be a *NP* hard problem. We recommend focus should now be given on working towards a graph theoretic solution for this problem with some simplifications. Such a solution will not only give maximum possible network information rate for a generic single source multisink multicast network but in the process it would also give the optimum switching strategy for the network.

Bibliography

[1] Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204-1216, July 2000.

[2] S.-Y. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. IT-49, no. 2, pp. 371-381, Feb. 2003.

[3] Christina Fragouli, JeanYves Le Boudec, Jorg Widmer, "Network Coding: An Instant Primer," *LCA-REPORT-2005-010*.

[4] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," *ISIT*, July 2003.

[5] R. W. Yeung, S.-Y. Li, N. Cai, and Z.Zhang,"Theory of network coding," submitted to *Foundations and Trends in Commun. and Inform. Theory*, preprint, 2005.

[6] C.K Ngai and R. W. Yeung, "Network coding gain of combination networks," *ITW*, Sant Antonio, Texas, pp. 283-287, 24-29 Oct. 2004.

[7] Xue-Bin Liang, "On the Switching Gap of Ahlswede-Cai-Li-Yeung's Single-Source Multicast Network," 2006 *IEEE Int. Symp. Inform. Theory*, Seattle, Washington, USA, July 2006.

[8] Xue-Bin Liang, "Matrix Games in the Multicast Networks: Maximum Information Flows With Network Switching," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, June 2006.

[9] G. Owen, *Game Theory*, 3^{rd} edition, San Diego: Academic Press, 1995.

[10] J. C. C McKinsey, *Introduction to the Theory of Games*. New York: McGraw-Hill Book Company, Inc., 1952, and Mineola, NY: Dover Publications, Inc., 2003.

[11] G.W Brown, "Iterative sollution of the games by fictious play." in *Activity analysis of production and Allocation*, T.C Koopmans, Ed. NewYork :Wiley, 1951, pp. 374-376.

[12] Dan Klein, "Lagrange Multipliers without Permanent Scarring," Computer Science Division, University of California at Berkeley. Available at www.cs.berkeley.edu/ klein/papers/lagrange-multipliers.pdf.

[13] R. J. McEliece, "Information multicasts," revised version of talk presented on Nov. 22, 2004 at *Lee Center meeting, California Institute of Technology, Pasadena, CA*. [online]. Available at http://www.ee.caltech.edu/EE/Faculty/rjm/papers/Lunch.pdf.

[14] J. Szep and F. Forgo, *Introduction to the Theory of Games*. Dordrecht: D. Reidel Publishing Company, 1985.

[15] N. Bhargava, "Switching Gap Analysis for generalized butterfly networks," *WPMC*, Jaipur, India, Oct 2007.

[16] P. Sanders, S. Egner, and L. Tolhuizen,"Polynomial time algorithms for network information flow," Proc. 15th ACM Symposium on Parallel Algorithms and Architectures, 2003.

[17] C. Fragouli and E. Soljanin. Decentralized network coding. Information Theory Workshop, Oct. 2004.

[18] "Avalanche: File swarming with network coding," http://research.microsoft.com/ pablo/avalanche.aspx.

[19] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," Infocom, Miami, FL, Mar 2005

[20] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard,"The importance of being opportunistic: Practical network coding for wireless environments," *Allerton*, 2005.

[21] D. Petrovic, K. Ramchandran, and J. Rabaey, "Overcoming untuned radios in wireless networks with network coding," *NetCod*, Italy, Apr. 2005.

[22] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," *IPSN*, Apr. 2005.

[23] C. Fragouli and A. Markopoulou, "A network coding approach to overlay network monitoring," *Allerton*, Sept. 2005.

[24] N. Cai and R. W. Yeung, "Secure network coding," *ISIT*, 2002.

[25] K. Bhattad and K. R. Narayanan, "Weakly secure network coding," *NetCod*, Apr. 2005.

[26] T. Ho, B. Leong, R. Koetter, M. Mdard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," *ISIT*, 2004.

[27] Thomas M. Cover and Joy A. Thomas, *Elements of Information Theory*, Wiley, 1991.

[28] Kamal Jain, Mohammad Mahdian and Mohammad Salavatipour,"Packing Steiner trees."

[29] Alexander Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer-Verlag, 2003.

[30] Ralf Koetter and Muriel Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol 11, pp. 782-795, Oct. 2003.

[31] Tracey Ho, Ralf Koetter, Muriel Medard, David R. Karger, and Michelle Effros, "The benefits of coding over routing in a randomized setting," *Proceedings of the IEEE International Symposium on Information Theory*, 2003.

[32] Sidharth Jaggi, Peter Sanders, Philip A. Chou, Michelle Effros, Sebastian Egner, Kamal Jain, and Ludo Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, submitted July 2003.

[33] Zongpeng Li, Baochun Li, Dan Jiang, and Lap Chi Lau, "On achieving throughput with network coding," *In Proceedings of INFOCOM 2005*, March 2005.

[34] Tracey Ho, "Networking from a network coding perspective," *PhD thesis*, MIT, 2004

[35] Chandra Chekuri, Christina Fragouli, and Emina Soljanin,"On Average Throughput and Alphabet Size in Network Coding," *IEEE Trans. Inform. Theory*, Vol. 52, No. 6, June 2006

[36] Dragon Petrovič, "Overcoming Untuned Radios in Wireless Networks With Network Coding,"*IEEE Trans. Inform. Theory*, Vol. 52, No.6, June 2006.

[37] Petar Maymounkov, Desmond S. Lun and Nicholas J. A. Harvey,"Methods for Efficient Network Coding."

[38] Desmond S. Lun, Muriel Médard, Tracey Ho, and Ralf Koetter,"Network Coding with a Cost Criterion," *Intl. Symp. Inform. Theory and its App.*, Parma, Italy, Oct 10-13, 2004.

[39] Peter Sanders, Sebastian Egner, Ludo Tolhuizen, "Polynomial Time Algorithms for Network Information Flow," *SPAA*, San Diego, California, USA, 2003.

[40] Yingda Chen, Shalinee Kishore and Jing (Tiffany) Li,"Wireless Diversity through Network Coding."

[41] Lihua Song, Raymond W. Yeung, and Ning Cai,"Zero-Error Network Coding for Acyclic Networks," *IEEE*, 2003.

[42] *Network Coding home page*, http://www.ifp.illinois.edu/ koetter/NWC/index.html

[43] Ashish Goel, "Algorithms for network routing, multicasting, switching and Design," *PhD thesis*, Stanford, 1999.

[44] Yunnan Wu, Philip A. Chou and Kamal Jain,"A Comparison of Network Coding and Tree Packing," *ISIT 2004*, Chicago, USA, June 27 July 2, 2004

[45] A. Agarwal and M. Charikar, "On the advantage of network coding for improving network throughput," *Proc. IEEE Infor. Theory Workshop*, San Antonio, TX, pp. 247-249, Oct. 2004.

[46] Umair F. Siddiqi, Sadiq M. Sait and Tarek R. Sheltami, "A New Algorithm to Find Multicast Routing Tree."

[47] James F. Kurose and Keith W. Ross, "Computer Networks, A top Down Approach Featuring the Internet," *Addison Wisely*, 2003.

[48] C. P. Low, X. Song, "On Finding Feasible Solutions for the Delay Constrained Group Multicast Routing Problem," *IEEE Trans. on Comp.*, vol. 51, No. 5, May 2002.

[49] C-W. Tsai, C-F. Tsai and C-P. Chen, "A Novel Multiple-Searching Genetic Algorithm for Multimedia Multicast Routing," *Proceeds. of the 2002 Congress on Evolutionary Computation*, 2002, Vol. 2, pp.1624-1629, 12-17 May 2002.

[50] Gang Feng, "A Heuristic for Multi-Constrained Multicast Routing," *Workshop on High Performance Switching and Routing*, 2004, pp. 309 - 313.

[51] Mohammed Aissa and Adil Ben Mnaouer, "A New Delay-constrained Algorithm for Multicast Routing Tree Construction," *IJCS*, 2004, vol. 17, pp. 985-1000.

[52] P. V. Mieghem and F.A. Kuipers, "Concepts of Exact QoS Routing Algorithms," *IEEE/ACM Trans. on Networking*, vol. 12, No. 5, October 2004.

[53] LIU Ying and WU Jianping, "A Genetic Algorithm for the Degree-Constrained Multicasting Problem," *5th IEEE International Conference on High Speed Networks and Multimedia Communications*, pp. 315-319, 3-5 July 2002.

[54] Y. Bartal, "Probabilistic aproximation of metric spaces and its algorithmic application," *37th IEEE Symp. on Foundation of Comp. Science,* pp. 184-196, 1996.

[55] Y.Bartal, "On approximating arbitrary metrics by tree metrics," *30th ACM Symposium on Theory of Computation*, 1998.

[56] M. Doar and I. Leslie, "How bad is naive multicast routing," *IEEE Infocom.*, pp. 82-89, 1992.

[57] D. Waitzman, C. Partridge and S. Deering, "Distance vector multicast routing protocol," *Internet RFC 1075*, 1988.

[58] M. Imase, B. Waxman, "Dynamic steiner tree problem," *SIAM J. Discrete Math.*, 4(3):369-384, August 1991.

[59] S. Deering, D.L. Estrin, D. Farinacci, C. Jacobson, V. Liu and L. Wei., "The pim architecture for wide-area multicast routing," *IEEE/ACM Trans. on Networking*, 4(2):153-152, April 1996.

[60] A. Ballardie, P.Francis and J. Crowcroft, "Core based trees (cbt) - an architecture for scalable inter-domain multicat routing," *ACM SIGCOMM*, pp. 85-95, 1993.

[61] S. McCanne, "Scalable compression and transmission of internet multicat video," *PhD Thesis*, Univ. of Calif. Berkley, 1996.

[62] M. Macedonia and D. Brutzman, "Mbone provides audio and video across the net," *IEEE Computer*, pp. 30-36, April 1994.

[63] L Kou, G Markowsky and L Berman, "A Fast Algorithm for Steiner trees," *Acta Informatica* Vol. 15, pp.141-145, 1981.

[64] http://www.personal.kent.edu/ rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/dijkstraAlg

[65] Z. Li, B. Li, D. Jiang, and L. C. Lau, "On achieving optimal end- to-end throughput in data networks: theoretical and empirical studies," *ECE Technical Report*, University of Toronto, 2004.