Secure network coding with a cost criterion

by

Jianlong Tan

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degrees of

Bachelor of Science in Electrical Engineering

and

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

[June 2006]

May 9, 2006

Author ....[illegible]........................
Department of Electrical Engineering and Computer Science
May 9, 2006

Certified by ....[illegible]........................
Muriel Médard
Associate Professor
Thesis Supervisor

Accepted by ...[illegible]..................
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Secure Network Coding with a Cost Criterion
by
Jianlong Tan

Submitted to the
Department of Electrical Engineering and Computer Science

May 9, 2006

In Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Electrical Engineering
and Master of Engineering in Electrical Engineering and Computer Science

Abstract

Network coding is a viable alterative to traditional routing methods that are being used in current data networks as it offers many advantages, including higher throughput rates and lower network costs. Although research has been conducted with regards to the cost and security issues of network coding, a joint investigation of these two parameters has not been done yet, thus providing the motivation for this thesis.

For this thesis, we consider the situation where a set of messages is to be multicasted across the network, of which a known subset is of interest to a wiretapping adversary. The problem that we attempt to solve is to find a network coding scheme that has both a low network cost and a low probability of the wiretapper being able to retrieve all the messages of interest.

We make use of random linear codes in anticipation for decentralized implementation of the scheme, and focus on the problem of finding the multicast subgraph. As an exact algorithmic solution is difficult, we propose two heuristic solutions, and compare their performances to traditional routing through a simulation study on Rocketfuel networks. Our results suggest that network coding can be more effective than routing for this low cost and secure data multicast problem, especially when the links are not easily tapped.

Thesis Supervisor: Muriel Médard

Title: Associate Professor, MIT Laboratory for Information and Decision Systems

Acknowledgements

I would like to thank my adviser Muriel Médard for her invaluable guidance and encouragement throughout the progress of this thesis, and my fellow coworkers in the Laboratory for Information and Decision Systems for their comments and help towards the completion of this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Network coding

In current communication networks, information is usually transmitted using a routing scheme, where data packets are sent from their places of origin to their destinations without being modified intentionally at their intermediate locations. As routing schemes are unable to always achieve the maximum information throughput of a network, Ahlswede et. al. [1] suggested a different data transfer scheme, now commonly known as "network coding," that can always achieve the maximum throughput through intentional modifications to the data packets within the network.

Consider the famous network coding example shown in Figure 1.1(a), where each directed edge has unit capacity. Suppose we need to transmit two bits of information, b_1 and b_2, from the source node s to each of the two sink nodes t_1 and t_2. Ahlswede et. al. showed that routing is unable to send these two data bits to their destinations in one single time step. Figure 1.1(b) shows an example of a routing scheme that is unable to send b_2 to t_1, primarily because the edge $3 \to 4$ has already been used to transmit b_1. However, network coding can be used to accomplish this data transfer as illustrated in Figure 1.1(c), where '+' denotes a modulo 2 addition. At t_1, b_2 can

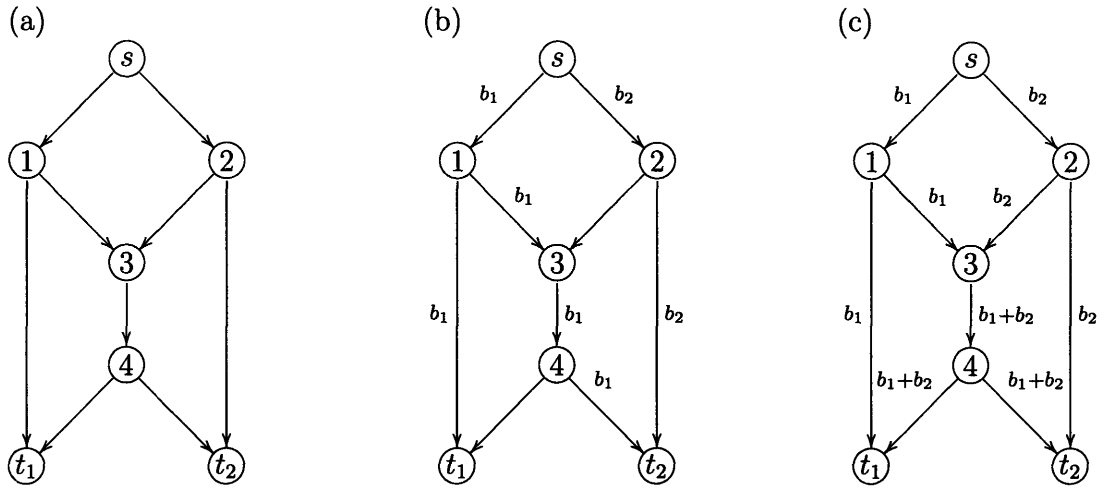

Figure 1.1: (a) A sample network. (b) A routing scheme where b_2 is unable to reach t_1. (c) A network coding scheme where both sink nodes can receive both messages.

be recovered from b_1 and $b_1 + b_2$ by taking their difference (modulo 2), and a similar argument follows at t_2.

The paper by Ahlswede et. al. triggered much research interest, and many papers have since been published for the better understanding of network coding. For instance, Li et. al. [2] proved that linear network codes over a large enough finite field is sufficient for multicast connections, while Ho et. al. [3] demonstrated that random linear network codes are able to support a data multicast with decreasing failure rates as the size of the finite field increases. In addition, Koetter and Médard [4] presented an algebraic framework to analyze network codes, and used it both to recapture past results and to show some new ones. For instance, they applied their framework to non-multicast situations and they also showed that the use of a single network code is sufficient even in a network with delays.

1.2 Cost concerns in network coding

Although network coding can achieve higher data throughput rates than routing, there are other aspects of communication that are important to network users.

One important performance parameter is the network cost incurred for a given set of connections. While the minimum cost multicast problem in routed networks requires the finding of a directed Steiner tree, which is NP-hard, the same problem in coded networks can be solved by a linear program in polynomial time [5], and also be implemented in a decentralized manner [6]. In addition, simulation results have shown that network coding can provide the multicast connections at a lower cost than traditional routing [6], [7].

1.3 Security concerns in network coding

Another important performance parameter is the security of the network. In this area, Cai and Yeung [8] considered the problem of using network coding to achieve perfect information security against wiretappers. In a separate paper, Feldman et. al. [9] showed that the finding of a matrix for the construction of an optimal secure network code is equivalent to the finding of a linear code with certain generalized distance properties. In a different setting, Ho et. al. [10] showed that randomized network coding is useful in detecting Byzantine modification of data packets, thus providing data security against Byzantine attackers who arbitrarily modify data packets in the network.

The remaining part of this section details the research work done by Cai and Yeung [8] and Feldman et. al. [9], as the notion of network security for this thesis is based primarily on their work.

In both papers, the researchers considered the problem of sending a single data stream X from its origin to its destinations, while ensuring *perfect information security* [16] against a wiretapping adversary. Mathematically, we say that perfect information security is achieved if, for any set of messages $\mathscr{Y}_{tapped}$ that is retrieved by the adversary, the conditional entropy of X given $\mathscr{Y}_{tapped}$, is the same as the entropy of X : (A short summary about information theory and entropy is given in section 2.4.)

$$H(X \mid \mathscr{Y}_{tapped}) = H(X).$$

In [8], it was assumed that a collection of sets of edges, $\mathscr{A} = \{A_1, \ldots, A_{|\mathscr{A}|}\}$ is known, and that the wiretapper can only tap the edges in one of its elements, A_i. Feldman et. al. [9] removed this assumption and generalized the problem to one where the wiretapper can tap at most ψ edges in the network.

In both papers, the authors achieved perfect information security by first encoding the data stream X with a random, independent data stream at the source node before transmission. Figure 1.2 shows an example of such a secure network code on the network discussed earlier, assuming that the wiretapper can tap at most one link in the network (i.e. $\psi = 1$). In the figure, b is a single bit message that is to be sent securely from s to t_1 and t_2, while w is the single bit encrypting message.

From the figure, it is easy to verify that the wiretapper can only retrieve one of the three types of messages sent in the network, namely w, $b + w$ and $b - w$. Since b and w are independent, it follows that:

$$H(b \mid w) = H(b).$$

In addition, similar to the situation where the one-time pad is able to provide perfect

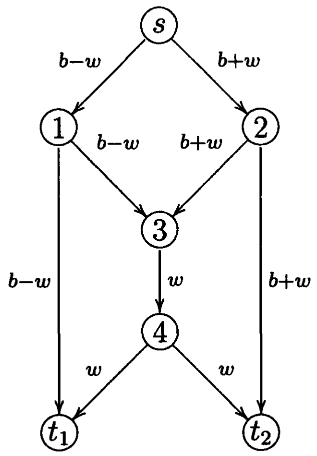

Figure 1.2: An example of a secure network code on the previous sample network, assuming that the wiretapper can access at most one link in the network. b denotes the secure message to be transmitted, while w denotes the encrypting message.

information security, we have:

$$\begin{aligned} H(b \mid b+w) &= H(b) \quad \text{and} \\ H(b \mid b-w) &= H(b). \end{aligned}$$

Thus, it follows that the network coding scheme shown in Figure 1.2 can achieve perfect information security against the wiretapper who can tap at most one network link.

1.4 Limitations to previous secure network coding scheme

Although the secure network coding scheme developed in [8] and [9] can provide perfect information security against the wiretapper, there are some limitations that we would like to point out and resolve in this thesis.

First, as it may be difficult to know the maximum number of links that are ac-

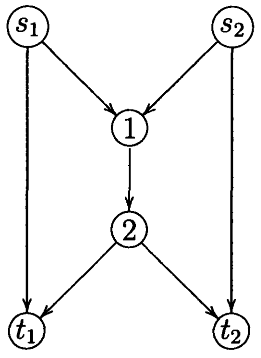

Figure 1.3: An example of a network where a secure network code does not exist. Suppose the message b is to be securely sent from s_1 to both t_1 and t_2, and that the wiretapper is capable of tapping any one link in the network. In this situation, no secure network code exists according to the previous schemes.

cessible to the wiretapper, we will instead assume that the wiretapper can tap each network link (i, j) with some probability p_{ij}, and that this set of probabilities is known for the whole network. This assumption is more realistic in application as it is usually easier to estimate or know this set of probabilities than the maximum number of links accessible to the wiretapper.

Second, although it is always desirable to have perfect information security, such a requirement may not always be necessary, or even possible. In some applications, it may be permissible to allow the wiretapper to receive some parts of the secure message. In other situations, such perfectly secure network codes may simply not exist.

Consider the network shown in Figure 1.3, where some message b is to be securely sent from s_1 to both t_1 and t_2, and that the wiretapper is capable of tapping any one link in the network. Since there is only one data path from s_1 to t_2, in order for t_2 to receive the message, the message b must be sent along the path $s_1 \to 1 \to 2 \to t_2$ in an uncoded manner. In such a situation, the wiretapper will always be able to retrieve the message b, as long as the tapped link lies on this critical data path.

With these considerations, instead of requiring that the wiretapper receive no information about the set of secure messages, we shall seek to achieve partial network security by trying to reduce the probability that the wiretapper is able to retrieve the full set of secure messages.

With this definition of network security, although the wiretapper can potentially receive a significant amount of information about the secure message, this does not mean that the security of the message is in serious jeopardy. This is because the purpose of this thesis is not to come up with a secure network coding scheme to replace the current standards of ensuring message security, but to come up with one that can further enhance message security. Thus, even if the wiretapper manages to retrieve all the necessary parts of the secure message, he will still need to overcome the other message security measures, such as cryptography and steganography. As a result, the aim of achieving partial network security is not a meaningless one.

Finally, as there are usually cost and throughput concerns associated with the issue of data transmission, it is desirable to address these concerns as well. For the previous secure network coding scheme, some random stream of encrypting data (e.g. bit w in Figure 1.2) must be transmitted across the network as well, which inevitably lowers the overall useful throughput of the network. In general, as the value of ψ increases, more encrypting data must be sent across the network, which inevitably lowers the throughput rate of useful information. This transmission of the encrypting data also increases the overall network cost, and is usually undesirable in situations where network resources are precious or scarce. Hence, while it is important to ensure a high level of security, there is also a need to maintain a low overall network cost.

1.5 Combining cost and security concerns in network coding

Having identified the limitations of the previous secure network coding scheme, we intend to come up with a partially secure network coding scheme that also tries to keep the network cost low. Thus, for this problem, two network parameters will be of primary interest, namely network cost and network vulnerability.

For this thesis, we shall consider the situation where a set of messages is to be multicasted across the network, of which a known subset is of interest to a wiretapper. The problem that we attempt to solve is to multicast the messages at a low cost, while keeping the network vulnerability — defined as the probability that the wiretapper is able to retrieve all the messages of interest — low. While network security is not limited to the resilience of the network against wiretapping, the other notions of security are beyond the scope of this work.

In general, we expect the existence of a tradeoff between network cost and network vulnerability, and thus, a joint minimization of both parameters is usually impossible. For instance, in routed networks, a cheapest cost approach to a unicast connection usually selects a single path. However, when the connection is to be resilient against wiretapping, multiple disjoint paths may be needed, which may increase the network cost [11], [12], [13], [14].

To illustrate further this trade-off and to show that network coding has the potential of achieving a lower network vulnerability than traditional routing, consider the network shown in Figure 1.5(a) where each network link has unit capacity and unit cost. Two random processes (denoted X_1 and X_2) are to be multicasted from the source nodes s_1 and s_2 to the sink nodes t_1 and t_2, against a wiretapper who is interested in obtaining the random process X_1. The probability that any one partic-

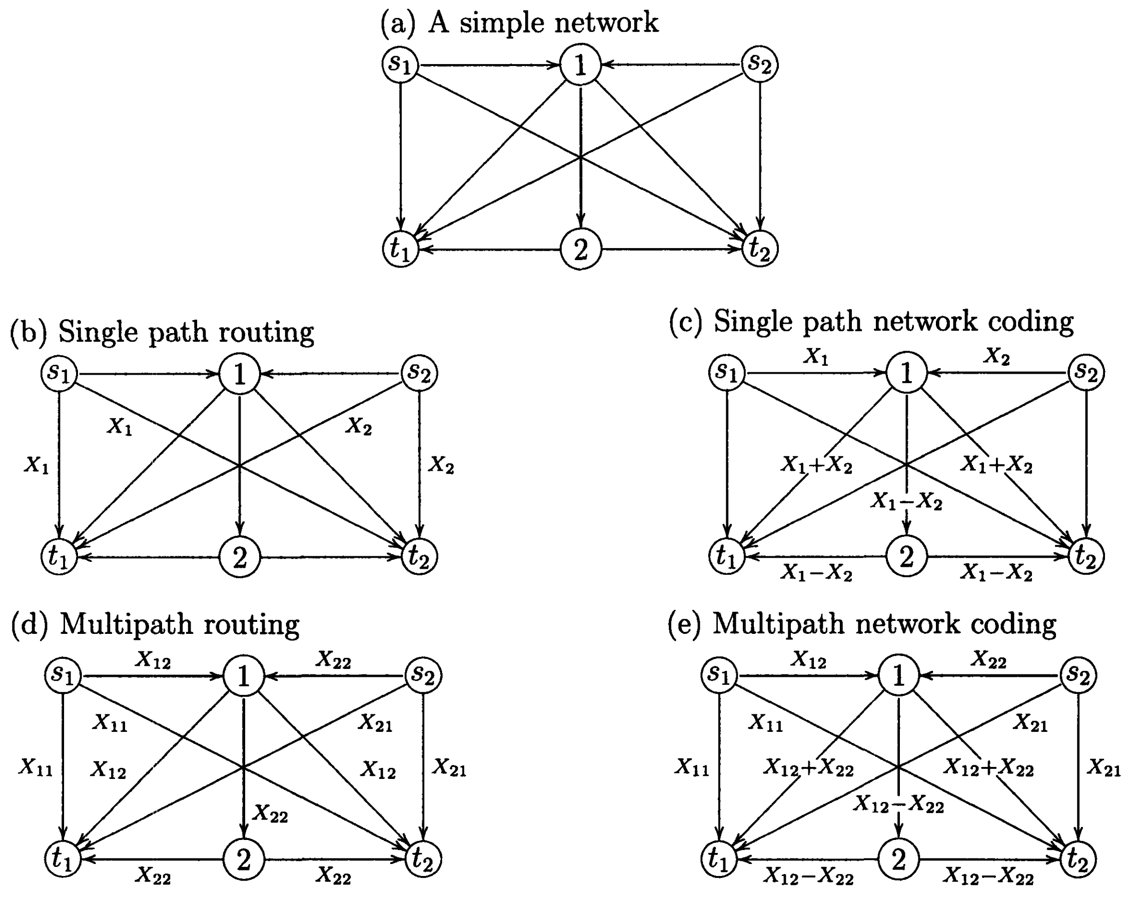

Figure 1.4: A simple network to illustrate the tradeoff between network cost and vulnerability.

ular link is tapped is 0.01, and edges are assumed to be tapped independently of one another.

Figures 1.5(b) to 1.5(e) show four different methods of achieving the multicast, and Table 1.1 shows the corresponding network costs and vulnerabilities. In Figures 1.5(b) and 1.5(c), each process is transmitted at unit rate, while in Figures 1.5(d) and 1.5(e), each process X_i is transmitted at a rate of two by splitting it into two processes X_{i1} and X_{i2}.

Figure 1.5(b) shows the single path routing solution that minimizes the network cost, while Figure 1.5(c) shows the non-trivial single path network coding solution

Table 1.1: Network costs and vulnerabilities for the simple network of Figure 1.5.

	Average cost (per bit)	Network Vulnerability
Single path routing	2	0.020
Single path coding	3.5	0.010
Multipath routing	2.75	5.9×10^{-4}
Multipath coding	2.75	2.1×10^{-5}

(note that the trivial solution is identical to the routing solution). Figure 1.5(d) shows the multipath routing solution and Figure 1.5(e) shows the multipath network coding solution.

From this simple example, we see that, while the single path routing solution offers the lowest network cost, it also results in the highest network vulnerability. While the network vulnerability can be reduced by employing network coding or multipath routing, the network cost inevitably increases. It should also be noted that the multipath network coding solution returns the lowest network vulnerability, at a cost equal to that of multipath routing.

1.6 Thesis outline

The organization for the rest of this thesis is as follows. Chapter 2 establishes the mathematical models to be used in this thesis, while Chapter 3 defines the problem to be solved in a mathematical framework and discusses the general solution to the problem. As the general solution is difficult, two heuristic approaches to the problem are presented in this chapter as well. Chapter 4 describes the setup used for the simulations and details the results obtained from them. Finally, a conclusion is given in Chapter 5, together with possible avenues for future work.

Chapter 2

Model

This chapter describes the various mathematical models used in this thesis. The first section expands on the network model presented in [5] for the finding of the coding subgraph. The second section describes the network model presented in [4] for the establishment of a linear network code.

As linear network codes are sufficient for multicast [2], and random linear network codes have the advantage of being implementable in a decentralized manner, we chose to make use of random linear codes. The random linear network coding model described in the original paper by Ho et. al. [3] is reproduced in the third section of this chapter.

The last section of this chapter describes information security under an information theoretic framework, and establishes the security criterion to be used henceforth.

2.1 Network model for finding the coding subgraph

A communication network is represented by a directed graph $G = (V, E)$, where V is the set of vertices (or nodes) and E is the set of directed edges (or links). With each edge $(i, j) \in E$, we associate two non-negative numbers c_{ij} and d_{ij}, which are the cost per unit flow and the link capacity, respectively. For simplicity, each edge is assumed to have infinite capacity ($d_{ij} \to \infty \; \forall \; (i,j) \in E$). We say that edge (i,j) is an *outgoing link* of node i and an *incoming link* of node j. In addition, we say that for the edge (i,j), node i is its *tail* and node j is its *head.*

In this network, a set of r discrete independent random processes $\mathscr{W} = \{W_1, \ldots, W_r\}$ is to be transmitted to a fixed set of sink nodes, $T = \{t_1, \ldots, t_{|T|}\} \subset V$. Each random process W_i is generated at the source node s_{W_i}, and is assumed to have a constant integer entropy rate of ρ. Thus, it can be arbitrarily well approximated by ρ independent random processes, each with unit entropy rate. We denote the j-th such component of W_i as the random process $X_{\rho(i-1)+j}$, and we denote the source node of process X_i as s_{X_i}. We let $\mathscr{X} = \{X_1, \ldots, X_{r\rho}\}$ be the set of mutually independent source processes with unit entropy rates.

In the network, there exists an adversary who is interested in obtaining all the messages in a given subset of $\mathscr{W}$. We denote this subset of messages as $\mathscr{W}_{interest}$ and assume without loss of generality that it contains the first k random processes (i.e. $\mathscr{W}_{interest} = \{W_1, \ldots, W_k\}$, $k \leq r$). In terms of the unit entropy rate source processes, we say that the adversary is interested in the processes in $\mathscr{X}_{interest} = \{X_1, \ldots, X_{k\rho}\}$.

This adversary is able to tap network links and retrieve all the messages that are being transmitted along them. To model this wiretapping behavior, we associate with each edge (i,j), a number $p_{ij} \in [0,1]$ to denote the probability that the wiretapper will be able to retrieve the message(s) sent along it. For simplicity, the tapping events of different network links are assumed to be independent (but the outputs of all tapped links are considered jointly when we consider the vulnerability of the network).

2.2 Network model for linear network codes

To aid in the description of the random linear network coding model in the following section, a different graph representation of the network is required. Specifically, we represent the coding subgraph as the graph $G' = (V, E')$, where each edge $l' \in E'$ has unit capacity, and links with greater capacities are modeled as parallel edges in E'.

In the network, edge l' carries the random process $Y(l')$, and each sink $t \in T$ receives a set of output random processes $\{Z(t,i)\}$. We denote the set of all transmitted random processes as $\mathscr{Y} = \cup_{l' \in E'}\{Y(l')\}$, and we denote the random set of all tapped messages as $\mathscr{Y}_{tapped} = \cup_{l' \in E'_{tapped}}\{Y(l')\} \subset \mathscr{Y}$, where $E'_{tapped} \subset E'$ is the random set of edges that are successfully tapped by the wiretapper. It is assumed that information is transmitted as vectors of bits of length u, represented as elements in the finite field $\mathbb{F}_{2^u}$.

For a delay-free network, we can represent each of the random processes X_i, $Y(l')$ and $Z(t,i)$ as a sequence of symbols from $\mathbb{F}_{2^u}$. Specifically, we can represent X_i as the sequence $X_i = \{X_i^{(0)}, X_i^{(1)}, \ldots\}$, $Y(l')$ as the sequence $Y(l') = \{Y^{(0)}(l'), Y^{(1)}(l'), \ldots\}$ and $Z(t,i)$ as the sequence $Z(t,i) = \{Z^{(0)}(t,i), Z^{(1)}(t,i), \ldots\}$.

For a network with delays, we represent it as the graph G' where every edge has unit time delay, using an appropriately chosen time unit. Links with greater delays are modeled as a series of edges in the graph (e.g. a link with a delay of t time units is represented as a series of t edges in the graph). Similar to the delay-free case, each of the random processes X_i, $Y(l')$ and $Z(t,i)$, can be represented by a sequence of symbols from $\mathbb{F}_{2^u}$.

2.3 Random linear network coding model

In anticipation of a completely distributed implementation of the secure network coding scheme, we make use of random linear network codes [3], which are not only sufficient for multicast [2], but are also easily implementable in a decentralized manner [6].

For a link l' with node v as its tail, we say that the signal $Y(l')$ is a random linear combination of the processes generated at v and the signals carried by the incoming links of v. In the case where all the links are delay-free, we can express $Y(l')$ as:

$$Y(l') = \sum_{\{i \in [1, r\rho]:\, s_{X_i} = v\}} a_{i,l'} X_i + \sum_{\{j' \in E':\, \text{head}(j') = v\}} f_{j',l'} Y(j'),$$

where $a_{i,l'}$ and $f_{j',l'}$ are uniformly and randomly chosen from the finite field $\mathbb{F}_{2^u}$.

Similarly, the output process $Z(t, i)$ at sink node t is a random linear combination of the signals carried by its incoming links, and can be expressed as follows in the delay-free case:

$$Z(t, i) = \sum_{\{l' \in E':\, \text{head}(l') = t\}} b_{t_{i,l'}} Y(l'),$$

where $b_{t_{i,l'}}$ is uniformly and randomly chosen from the finite field $\mathbb{F}_{2^u}$.

For networks with link delays, memory is needed at the receiver nodes, but memoryless operations at all other nodes will suffice [4]. The symbol of $Y(l')$ at time $\tau + 1$, denoted as $Y^{(\tau+1)}(l')$, is a random linear combination of the symbols that are accessible at node $v = \text{tail}(l')$ one time step earlier at time τ. Mathematically, we can express $Y^{(\tau+1)}(l')$ as:

$$Y^{(\tau+1)}(l') = \sum_{\{i \in [1, r\rho]:\, s_{X_i} = v\}} a'_{i,l'} X_i^{(\tau)} + \sum_{\{j' \in E':\, \text{head}(j') = v\}} f_{j',l'} Y^{(\tau)}(j').$$

At the sink nodes, if a memory of m is made available for each output process at the sink nodes, we can then express the symbol of $Z(t,i)$ at time $\tau+1$ as:

$$Z^{(\tau+1)}(t,i) = \sum_{u=0}^{m} b'^{(u)}_{t_{i,l'}} Z^{(\tau-u)}(t,i) + \sum_{\{l'\in E' \colon \text{head}(l')=t\}} \sum_{u=0}^{m} b''^{(u)}_{t_{i,l'}} Y^{(\tau-u)}(l'),$$

where $b'^{(u)}_{t_{i,l'}}$ and $b''^{(u)}_{t_{i,l'}}$ are uniformly and randomly chosen from the finite field $\mathbb{F}_{2^u}$.

These equations can be represented algebraically in terms of a delay variable D as follows:

$$\begin{aligned} Y(l')(D) &= \sum_{\{i\in[1,r\rho] \colon s_{X_i}=v\}} a_{i,l'} X_i(D) + \sum_{\{j'\in E' \colon \text{head}(j')=v\}} D f_{j',l'} Y(j')(D), \\ Z(t,i)(D) &= \sum_{\{l'\in E' \colon \text{head}(l')=t\}} b_{t_{i,l'}} Y(l')(D), \end{aligned}$$

where:

$$\begin{aligned} a_{i,l'} &= D a'_{i,l'}, \\ b_{t_{i,l'}} &= \frac{\sum_{u=0}^{m} D^{u+1} b''^{(u)}_{t_{i,l'}}}{1 - \sum_{u=0}^{m} D^{u+1} b'^{(u)}_{t_{i,l'}}}, \end{aligned}$$

and

$$\begin{aligned} X_i(D) &= \sum_{\tau=0}^{\infty} X_i^{(\tau)} D^{\tau}, \\ Y(l')(D) &= \sum_{\tau=0}^{\infty} Y^{(\tau)}(l') D^{\tau}, & Y^{(0)}(l') = 0, \\ Z(t,i)(D) &= \sum_{\tau=0}^{\infty} Z^{(\tau)}(t,i) D^{\tau}, & Z^{(0)}(t,i) = 0. \end{aligned}$$

The coding coefficients $\{a_{i,l'}, f_{j',l'}, b_{t_{i,l'}} \in \mathbb{F}_{2^u}\}$ can be consolidated into $r\rho \times |E'|$

matrices $A = (a_{i,l'})$ and $B_t = (b_{t_{i,l'}})$, and the $|E'| \times |E'|$ matrix $F = (f_{j',l'})$. The matrix A can be thought of as the transfer matrix from the source processes X_i to the link processes $Y(l')$. Similarly, the matrix B can be thought of as the transfer matrix from the link processes $Y(l')$ to the output processes $Z(t,i)$.

The triple (A, F, B) where:

$$B = \begin{bmatrix} B_{t_1} \\ --- \\ \vdots \\ --- \\ B_{t_{|T|}} \end{bmatrix},$$

uniquely defines the linear network code used for the network. From the results in [4], the connection problem is said to be feasible with the triple (A, F, B) if and only if the matrix AMB_t^T has full rank for each receiver t, where:

$$M = \begin{cases} (I-F)^{-1} = I + F + F^2 + \dots & \text{in the case without delays,} \\ (I-DF)^{-1} = I + DF + D^2F^2 + \dots & \text{in the case with delays.} \end{cases}$$

2.4 Information security model

For a random variable $X^{(\tau)}$, its *entropy* $H(X^{(\tau)})$, can be computed as follows:

$$H(X^{(\tau)}) = -\sum_x p_x \log_2 p_x,$$

where p_x denotes the probability that the random variable $X^{(\tau)}$ takes on the value x.

For each random process $X_i = \{X_i^{(0)}, X_i^{(1)}, \dots, X_i^{(|X_i|)}\}$, we can approximate it arbitrarily well by a sequence of independent and identically distributed (i.i.d.) symbols, because compression can be done at the source nodes to reduce redundancy in the transmitted sequence. As a result, the entropy of the random process X_i can be expressed as:

$$H(X_i) = \sum_{\tau} H(X_i^{(\tau)}).$$

Since it was assumed that the random processes in $\mathscr{X} = \{X_1, \ldots, X_{r\rho}\}$ are mutually independent, we can express the conditional entropy of X_i, given X_j, as follows:

$$H(X_i \mid X_j) = \begin{cases} H(X_i) & \text{if } i \neq j, \\ 0 & \text{if } i = j. \end{cases}$$

Then, for the set of messages of interest to the wiretapper, its entropy is defined as:

$$\begin{aligned} H(\mathscr{X}_{interest}) &= \sum_{\{X_i \in \mathscr{X}_{interest}\}} H(X_i) \\ &= \sum_{i=1}^{k\rho} H(X_i). \end{aligned}$$

Now, suppose the wiretapper obtains some of the messages that are sent in the network. Denoting this set of tapped messages as $\mathscr{Y}_{tapped}$, we say that the conditional entropy of $\mathscr{X}_{interest}$ given $\mathscr{Y}_{tapped}$ is $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})$.

In effect, this conditional entropy provides us the answer to the question: "Given that the wiretapper has retrieved all the messages in $\mathscr{Y}_{tapped}$, how much information about the messages in $\mathscr{X}_{interest}$ is he still missing?" In order for the wiretapper to retrieve the full identities of all the messages in $\mathscr{X}_{interest}$, he will need to make a guess out of $2^{H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})}$ possibilities.

The next thing to note about the conditional entropy $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})$ is that it is always a non-negative value, and it can never exceed the entropy value of $\mathscr{X}_{interest}$. Thus, the following inequality holds:

$$0 \leq H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped}) \leq H(\mathscr{X}_{interest}).$$

From the interpretation of the conditional entropy, it follows that as the value of $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})$ increases, the level of network security increases, as the wiretapper needs to make a guess out of an increasing number of possibilities. However, from the above inequality, it follows that the conditional entropy is at most equal to $H(\mathscr{X}_{interest})$. As the set of tapped messages is random, we say that perfect information security is achieved when the following equation holds for any realization of the set $\mathscr{Y}_{tapped}$:

$$H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped}) = H(\mathscr{X}_{interest}).$$

This notion of perfect information security is the aim of the papers [8] and [9], and as explained earlier, it may be too strict in application, and thus, a weaker security criterion is used for this thesis. Specifically, we consider the worst case scenario, whereby the conditional entropy $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})$ is equal to zero, which is equivalent to saying that the wiretapper is able to retrieve all the messages in the set $\mathscr{X}_{interest}$ without having to make any arbitrary guesses. Since $\mathscr{Y}_{tapped}$ is a random set, we define the network vulnerability as the probability that the worst case scenario happens:

$$\nu := \Pr[H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped}) = 0]. \qquad (2.1)$$

With this weaker notion of network security, there are times when the adversary is able to obtain some, or even all, of the messages of interest.

Given a network code (A, F, B), and a fixed realization of the random set E'_{tapped}, we say that the wiretapper is able to fully retrieve all the k messages of interest if and only if there exists a $r\rho \times |E'|$ matrix $B_{tapped} = (b_{i,l'})$ where:

1. $b_{i,l'} \in \mathbb{F}_{2^u} \quad \forall\, i \in [1, r\rho],\ l' \in E'$.
2. $b_{i,l'} = 0 \quad \forall\, l' \notin E'_{tapped}$.
3. The $r\rho \times r\rho$ matrix $\Omega = AMB^T_{tapped} = (\omega_{i,j})$ can be represented in the block form:

$$\Omega = \begin{bmatrix} \Omega_{1,1} & \Omega_{1,2} \\ \Omega_{2,1} & \Omega_{2,2} \end{bmatrix},$$

where the submatrices have the following properties:

   (a) $\Omega_{1,1}$ has dimensions $k\rho \times k\rho$, and is invertible.

   (b) $\Omega_{1,2}$ has dimensions $k\rho \times (r-k)\rho$, and is a null matrix.

   (c) $\Omega_{2,1}$ has dimensions $(r-k)\rho \times k\rho$, and is a null matrix.

   (d) $\Omega_{2,2}$ has dimensions $(r-k)\rho \times (r-k)\rho$, and is a null matrix.

Although the notion of network security can be strengthened by defining the network vulnerability as the probability that the conditional entropy $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped})$ exceeds a certain threshold amount, such a definition will inevitably complicate the computation of the network vulnerability.

Since there are $k\rho$ processes of interest in $\mathscr{X}_{interest}$, if we were to say that network security is compromised when m or more of these $k\rho$ processes are fully retrieved, then it follows that the total number of scenarios that will need to be considered is $\sum_{i=m}^{k\rho} \binom{k\rho}{i}$, which generally scales exponentially with $k\rho$. Thus, for simplicity, we only concern ourselves with the single scenario where all the messages are compromised (i.e. $H(\mathscr{X}_{interest} \mid \mathscr{Y}_{tapped}) = 0$).

Chapter 3

Problem Formulation and Solution

This chapter first defines the problem to be solved in this thesis, before making a remark about the previous secure coding scheme described in section 1.3. Following this, the third section of this chapter describes the general solution to the problem, and discusses some of the difficulties associated with it. As the general solution is difficult, heuristics are needed for the solving of the problem and two heuristics to the problem are discussed in the last two sections.

3.1 Problem formulation

As mentioned earlier, this thesis is concerned with the problem of coming up with a secure network coding scheme that takes into account both network cost and network vulnerability.

For the network cost, we consider the graph $G = (V, E)$ where each edge $(i, j) \in E$ has a cost per unit flow of c_{ij}. Thus, if z_{ij} denotes the amount of information flow on the edge $(i, j) \in E$, it then follows that the total network cost is given by:

$$\mu \ := \sum_{\{(i,j) \in E\}} c_{ij} z_{ij}.$$

For the network vulnerability, as mentioned in section 2.4, we define the network vulnerability as the probability that the wiretapper is able to retrieve all the messages in the set $\mathscr{W}_{interest}$ (or equivalently, in terms of the unit entropy rate source processes: $\mathscr{X}_{interest}$). Mathematically, the network vulnerability is given by Equation 2.1.

With the network cost and vulnerability defined as such, the problem to be solved is to minimize some function $\mathcal{F}(\mu, \nu)$, which is assumed to be an increasing function of both μ and ν. Once again, as we expect a trade-off between network cost and network vulnerability, it is often inadequate to minimize only μ or ν.

3.2 Using other messages in the network to provide security

From the previous secure coding scheme described in section 1.3, we observe that information security is achieved through two main steps. First, the secure message has to be encrypted by a random, independent message at the source. Second, the messages to be transmitted on the network links themselves must be such that the wiretapper does not obtain any information about the secure message from the network links that he can tap.

On closer examination, one will notice that this source encryption is very similar to the concept of network coding. Returning to Figure 1.2, we see that the secure message b is encrypted with a random, independent message w that is generated at the source node s. Now, suppose that the network shown in Figure 1.2 is just a section of a larger network, and let us further suppose that w is not a random data stream generated at s, but is, instead, another message that is to be transmitted in the network via the network node s. With these assumptions, the network code shown in Figure 1.2 will still apply, and perfect information security against the wiretapper is achieved in this section of the network.

This similarity between source encryption and the mixing of messages in network coding shows that data security can also be achieved by using other messages in the network as encrypting messages. The main advantage of using other messages for encryption is that there is no longer a need to transmit random streams of encrypting data that lower useful network throughput and increase the total network cost.

For this thesis, as network cost is of significant importance, we ignore the possibility of increasing network security at the expense of increased network cost by introducing random data streams into the network. Thus, we restrict ourselves to the minimization of the function $\mathcal{F}(\mu, \nu)$ without introducing additional messages into the network.

Here, it should be noted that despite restricting ourselves to the minimization of $\mathcal{F}(\mu, \nu)$ without introducing additional messages, one can still investigate the effect of introducing random data streams into the network by keeping the set of messages of interest $\mathscr{X}_{interest}$ fixed, while increasing the size of the set of source processes $\mathscr{X}$.

3.3 General solution

We separate our secure coding problem into two parts. The first part deals with the finding of the coding subgraph, including the amount of information flow to be put onto each network link. The second part involves the actual network code to be implemented in the coding subgraph. The network code describes how data packets interact with one another, and is important to ensure that the original messages can be decoded at the sink nodes.

In anticipation of a distributed implementation of the solution, we make use of random linear network codes, which are sufficient for multicast [2]. (The random coding model is described in section 2.3.) With this decision, we turn our attention to the problem of finding the coding subgraph in $G = (V, E)$.

Since every sink node in T needs to receive all the random processes in $\mathscr{W}$, we can translate this multiple multicast problem into a single multicast one by introducing a pseudo-source node α, and r pseudoedges (α, s_{W_i}) into the graph G. The pseudo-source node generates all the processes in $\mathscr{W}$ and transmits them to the actual source nodes $s_{W_1}, \dots, s_{W_r}$ via the pseudoedges (α, s_{W_i}) that have zero cost per unit flow (i.e. $c_{ij} = 0, \ \forall\, i = \alpha$).

Noting that the problem of finding the minimum cost subgraph for a single multicast connection can be cast as a linear programming problem [5], we structure the subgraph finding problem in a similar manner as the following constrained optimization, where z_{ij} denotes the actual amount of information flow on edge (i, j) and $x_{ij}^{(t)}$ denotes the underlying amount of information flow on edge (i, j) that is useful to sink node t:

$$
\begin{aligned}
&\text{minimize} \quad \mathcal{F}(\mu, \nu) \\
&\text{subject to} \quad z_{ij} = \rho, && \forall\, i = \alpha, \\
&z_{ij} \geq x_{ij}^{(t)}, && \forall\, (i,j) \in E,\ t \in T, \\
&\sum_{\{(i,j)\in E\}} x_{ij}^{(t)} - \sum_{\{(j,i)\in E\}} x_{ji}^{(t)} = \sigma_i^{(t)}, && \forall\, i \in V,\ t \in T, \\
&d_{ij} \geq x_{ij}^{(t)} \geq 0, && \forall\, (i,j) \in E,\ t \in T,
\end{aligned}
\tag{3.1}
$$

where:

$$
\begin{aligned}
\mu &= \sum_{\{(i,j)\in E\}} c_{ij} z_{ij}, \\
\nu &= \Pr\left[H\left(\mathscr{W}_{interest} \mid \mathscr{Y}_{tapped}\right) = 0\right], \\
\sigma_i^{(t)} &= \begin{cases} r\rho & \text{if } i = \alpha, \\ -r\rho & \text{if } i = t, \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}
$$

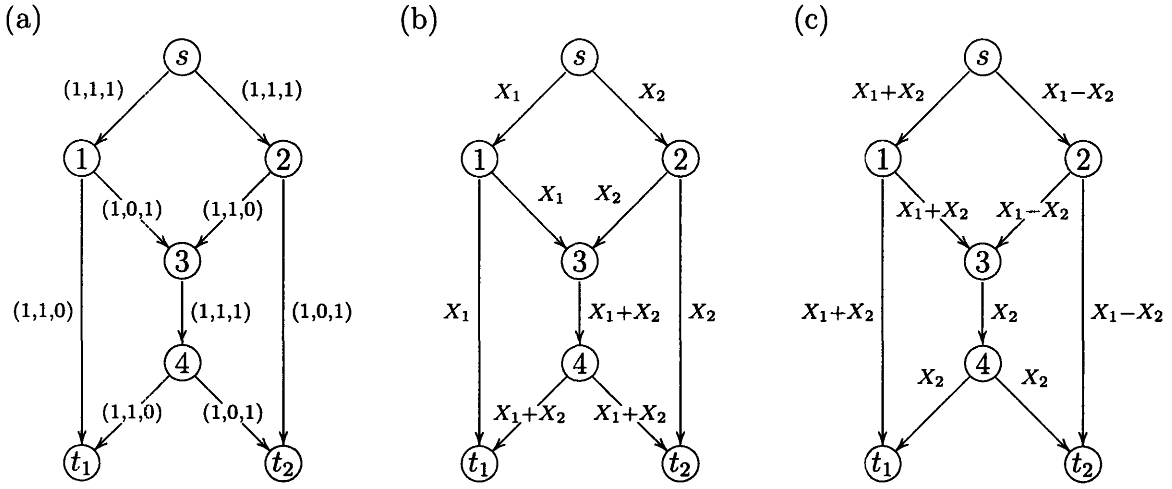

Figure 3.1: Example showing that network vulnerability depends on the network code. (a) Values of $(z_{ij}, x_{ij}^{(t_1)}, x_{ij}^{(t_2)})$. (b) Network code I. (c) Network code II.

One difficulty of this optimization is that, while it is easy to compute μ given $x_{ij}^{(t)}$ and z_{ij}, the same is not true for ν, as it depends on the actual processes sent along the links (i.e. the network code).

As an example, consider the network shown in Figure 3.1, with $\rho = 1$, $\mathscr{W} = \{W_1, W_2\} = \{X_1, X_2\}, \mathscr{W}_{interest} = \{W_1\} = \{X_1\}$. Figure 3.1(a) shows the values of $x_{ij}^{(t)}$ and z_{ij}, while Figures 3.1(b) and 3.1(c) show two possible network codes utilizing the values of $x_{ij}^{(t)}$ and z_{ij} shown in Figure 3.1(a).

Consider the network code used in Figure 3.1(b). Here, we see that the tapping of any one edge in the set $\{(s,1),(1,3),(1,t_1)\}$ will allow the retrieval of the message in $\mathscr{X}_{interest}$. However, for the network code in Figure 3.1(c), this message can only be fully retrieved when the wiretapper taps two or more edges carrying different messages.

For the sake of illustration, let us calculate the network vulnerabilities for the two networks shown in Figures 3.1(b) and (c), assuming that $p_{ij} = p = 0.01 \; \forall \; (i,j) \in E$. Then, we let $\overline{X_i}$ denote the event that message X_i is not tapped from the network, and $\overline{\overline{X_i}}$ denote the event that it is successfully tapped. Denoting the vulnerabilities of the two networks in Figures 3.1(b) and (c) as $\nu_{(b)}$ and $\nu_{(c)}$ respectively, it follows that:

$$\begin{aligned}
\nu_{(b)} &= 1 - \Pr\left(\overline{X_1}\right) \cdot \left[1 - \Pr\left(\overline{\overline{X_2}}\right) \cdot \Pr\left(\overline{\overline{X_1 + X_2}}\right)\right] \\
&= 1 - (1-p)^3 \cdot [1 - (1 - (1-p)^3) \cdot (1 - (1-p)^3)] \\
&= 0.0306. \\
\nu_{(c)} &= 1 - \Big[\Pr\left(\overline{X_1 + X_2}\right) \cdot \Pr\left(\overline{X_1 - X_2}\right) \cdot \Pr\left(\overline{X_2}\right) \\
&\qquad + \Pr\left(\overline{\overline{X_1 + X_2}}\right) \cdot \Pr\left(\overline{X_1 - X_2}\right) \cdot \Pr\left(\overline{X_2}\right) \\
&\qquad + \Pr\left(\overline{X_1 + X_2}\right) \cdot \Pr\left(\overline{\overline{X_1 - X_2}}\right) \cdot \Pr\left(\overline{X_2}\right) \\
&\qquad + \Pr\left(\overline{X_1 + X_2}\right) \cdot \Pr\left(\overline{X_1 - X_2}\right) \cdot \Pr\left(\overline{\overline{X_2}}\right)\Big] \\
&= 1 - [(1-p)^9 + 3 \times (1-p)^6 \times (1 - (1-p)^3)] \\
&= 0.0026 \neq \nu_1.
\end{aligned}$$

Thus, we see that the network vulnerability depends not only on the values of $x_{ij}^{(t)}$ and z_{ij}, but also on the actual network code.

Another difficulty is that ν and $\mathcal{F}(\mu, \nu)$ may not be convex functions of $x_{ij}^{(t)}$ and z_{ij}. For instance, let us return to the network shown in Figure 3.1(b), where we now assume that every edge has a capacity of two. In addition, for each edge in the set $\{(s,1), (1,t_1), (s,2), (2,t_2)\}$, we now transmit a new message X_3 in an uncoded fashion, in addition to the original message that it has been transmitting. As the knowledge of X_3 does not help the wiretapper in knowing the identity of X_1, the increases in z_{ij} for these edges do not affect ν.

As a result of these difficulties with the general solution, we turn towards heuristic methods to solve the problem.

3.4 First heuristic

Our first heuristic is motivated by the special scenario where all the input processes are generated at a single source node. In this scenario, there is a high probability that all the processes in $\mathscr{Y}$ will be a linear combination of all $r\rho$ input processes of $\mathscr{X}$. Thus, in order for the wiretapper to retrieve the processes in $\mathscr{X}_{interest}$, there is a high probability that he will need to retrieve all $r\rho$ degrees of freedom in order to invert the coding matrix.

In this light, we formulate our first heuristic, which is an exact solution to the problem when the following statements are true.

1. Every transmitted process in the network is a linear combination of all $r\rho$ processes in $\mathscr{X}$. Mathematically, we have:

$$Y(l') = \sum_{i=1}^{r\rho} \zeta_{l',i} X_i, \qquad \forall\, l' \in E',$$

where:

$$\zeta_{l',i} \in \mathbb{F}_{2^u} \backslash \{0\}, \qquad \forall\, l' \in E',\ 1 \leq i \leq r\rho.$$

2. The tapping of any n links in E' by the wiretapper will provide him with a set of $\min(r\rho, n)$ linearly independent equations in terms of the processes in $\mathscr{X}$.

While these two statements are not true in general, they can be reasonable approximations to the actual network code when the processes are well-mixed throughout the network, and when the finite field used for coding is very big.

As mentioned in the previous section, one of the difficulties associated with the general solution is that the network vulnerability is not a function of only $x_{ij}^{(t)}$ and z_{ij}, but is dependent on the actual network code as well. However, when the above two conditions hold, the network vulnerability is equal to the probability that the wiretapper manages to tap at least $r\rho$ links in E', which is a function of the z_{ij} values only (and is independent of the actual network code). This probability, denoted as ν', can be obtained in the following manner.

Consider a link $(i,j) \in E$ that is carrying a flow amount $z_{ij} \in \mathbb{R}_0^+$. Since each link in E' has unit capacity, the number of unit capacity links in E' that correspond to the link (i,j) is approximately $[z_{ij}]$ — the nearest integer to z_{ij}. With this, we let L_{ij} be the discrete random variable denoting the amount of information flow successfully tapped on link (i,j). As p_{ij} is the probability of tapping link (i,j), we have:

$$\Pr(L_{ij} = k) = \begin{cases} p_{ij} & \text{if } k = [z_{ij}], \\ 1 - p_{ij} & \text{if } k = 0. \end{cases}$$

To avoid confusion in the z-transform domain, we replace the transform variable z by w, and we write:

$$\begin{aligned} L_{ij}(w) &= E[w^{L_{ij}}] \\ &= p_{ij} w^{[z_{ij}]} + (1 - p_{ij}). \end{aligned}$$

Now, we let L_{total} be the random variable denoting the total amount of information flow successfully tapped in the whole network. Since edges are assumed to be tapped independently of one another, we have:

$$\begin{aligned} L_{total}(w) &= \prod_{(i,j)\in E} L_{ij}(w) \\ &= \sum_k \Pr(L_{total} = k) w^k, \end{aligned}$$

and the expression for ν' is:

$$\begin{aligned}\nu' &= \Pr(L_{total} \geq r\rho) \\ &= 1 - \sum_{k=0}^{r\rho-1} \Pr(L_{total} = k),\end{aligned}$$

which is an increasing function with respect to each z_{ij}, but may not be convex in nature.

Algorithm 1 below summarizes how to obtain ν' from the vector $z = (z_{ij})$. In the algorithm, k_{ij} is the vector of polynomial coefficients of $L_{ij}(w)$ and k_{total} is the vector of polynomial coefficients of $L_{total}(w)$, which is obtained by convolving all the k_{ij} vectors. Note that the symbol $*$ is used to denote the convolution operation.

```
$z'_{ij} \leftarrow \lceil z_{ij} \rceil \quad \forall\ (i,j) \in E$;
for each $(i,j) \in E$ do
    $k_{ij} \leftarrow$ zero vector of length $(z'_{ij} + 1)$;
    if $z'_{ij} = 0$ then
        $k_{ij}[1] \leftarrow 1$;
    else
        $k_{ij}[1] \leftarrow p_{ij}$;
        $k_{ij}[z'_{ij} + 1] \leftarrow 1 - p_{ij}$;
    end
end
$k_{total} \leftarrow \underset{\{(i,j)\in E\}}{*} k_{ij}$;
$h \leftarrow length(k_{total})$;
$\nu' \leftarrow 1 - \sum_{j=h-r\rho+1}^{h} k_{total}[j]$;
```

Algorithm 1: Given z, calculate ν'.

With this, we replace the objective function in (3.1) by the function $\mathcal{F}(\mu, \nu')$, which is a function in terms of z_{ij} only. For our simulations, we consider a specific form of $\mathcal{F}(\mu, \nu')$, given by:

$$\mathcal{F}(\mu, \nu') = \mu + \omega\nu', \tag{3.2}$$

where ω is a non-negative weighting variable that represents the relative importance of the network vulnerability with respect to the overall network cost.

3.5 Second heuristic

One major drawback of the first heuristic is that the calculation of ν' can be computationally intensive, as it requires the calculation of the polynomial coefficients of $L_{total}(w)$. Because of this complexity, a simpler heuristic is proposed here.

We begin by drawing some insights from the way routed networks provide resilience against wiretapping. In traditional routing schemes, when a message needs to be securely transmitted to its destinations, a dispersive routing scheme is often used. Aside from providing better information security, other advantages of such a dispersive routing scheme have also been well investigated in a number of papers [11], [12], [13], [14].

In such a routing scheme, the secure message will be broken into smaller parts and all these parts will be sent to the destinations of the original message through several disjoint paths. Security against wiretapping is increased through the use of these disjoint paths because, in order for the wiretapper to retrieve the entire message, the wiretapper will need to tap a link on each path used for data transmission. Thus, as the number of disjoint paths used increases, the wiretapper will need to tap more links in order to retrieve the original message, and he will have a decreasing probability of being able to do so.

As the spreading of information flow across disjoint paths was shown to provide increased data security against wiretapping in traditional routed networks, it is believed that is will also provide increased data security in network coding by similar principles.

For instance, if all the flows are concentrated along a single multicast tree in a coded network, then $\Pr(L_{total} \geq r\rho)$ can be quite high, as each used link in the network is carrying a large amount of flow, and one only needs to tap a few of them to have $L_{total} \geq r\rho$. However, if the flows are spread out across network links instead, one will need to tap more network links to have $L_{total} \geq r\rho$. Hence, we expect the network vulnerability to decrease as information flow is spread across the entire network.

A straightforward method to spread information flow across network links is to introduce, for each link $(i, j) \in E$, a strictly convex and increasing cost function in terms of z_{ij}. For our simulations, we consider the addition of a quadratic cost function to each link. Specifically, the objective function in (1) is replaced by the following function:

$$\mu + \omega' \mu_{quad}, \tag{3.3}$$

where:

$$\mu_{quad} = \sum_{\{(i,j) \in E\}} p_{ij} \left(z_{ij}/\rho\right)^2,$$

and ω' is a non-negative weighting variable that represents the relative importance of the network vulnerability with respect to the overall network cost. Here, it is important to note that $\omega \neq \omega'$ in general, because $\nu' \leq 1$, while $\mu_{quad} \leq r^2$. As it is much easier to compute μ_{quad} than ν', this heuristic is easier to implement than the previous one. Another advantage that this heuristic has is that it can be implemented in a fully distributed manner as each edge has a strictly convex cost function [6].

Chapter 4

Simulation Results

This chapter details the results obtained from our simulations. The simulation setup is explained in the first section, before the results and discussions are presented in the remaining ones. The second section details the general results obtained across different networks, while the third section discusses the tradeoff relationship between network cost and network vulnerability. The remaining sections of this chapter discuss the effects of the different network parameters on the performances of the proposed heuristics.

4.1 Simulation setup

Our simulations are done using the network topologies made available by the Rocketfuel project [15]. For our simulations, we make the simplifying assumption that all network links have the same probability of being tapped ($p_{ij} = p$, $\forall\ (i,j) \in E$).

For each scenario, we first decide on the network parameters, including the number of input processes to be transmitted (r), the throughput rate for each process (ρ), the number of processes of interest to the wiretapper (k) and the number of sink nodes ($|T|$). We also decide on the network to be used, which gives us the set of c_{ij} values.

We then proceed to run 100 simulations with the above network parameters, and calculate the mean network cost and network vulnerability. For each simulation, each source node s_{W_i} is randomly and uniformly chosen from V with replacement, before each sink node in T is randomly and uniformly chosen from $V - S$ without replacement. During this process, we also ensure that the multicast problem is feasible, by requiring the presence of at least one path from each source node to each sink node.

Following this, we let p vary from 0 to 0.1, and for each value of p, we proceed to solve for five different multicast subgraphs. We first consider the single path routing and the multipath routing solutions, before considering the single path network coding solution by having the objective function in (3.1) as μ. Finally, we consider the two heuristics proposed in the previous chapter.

The objective function to be minimized for each simulation is a weighted sum of network cost and network vulnerability. In particular, the form of $\mathcal{F}(\mu, \nu)$ that we consider is as follows:

$$\mathcal{F}(\mu, \nu) \quad = \quad \mu + \omega\nu,$$

where ω is a non-negative weighting variable. As for the two proposed heuristics, the corresponding objective functions have already been given in the previous chapter in Equations 3.2 and 3.3.

For each subgraph, we estimate the network vulnerability in the following way. We randomly generate 2000 sets of tapped links based on p, and decide for each set, if it enables the wiretapper to retrieve all the processes in $\mathscr{W}_{interest}$. The sample mean is then taken to be the network vulnerability ν.

4.2 General results for various networks

We first consider the applicability of our heuristics across different networks. Specifically, we fix the values for the network parameters r, k, ρ and $|T|$, and obtain the simulation results for four different networks. The four networks used in the simulations are the Ebone (Europe), Exodus (USA), Telstra (Australia) and Tiscali (Europe) networks. Although two other network topologies were available from the Rocketfuel project, they were not used in the simulations. The Sprint network (USA) was too large for the simulations to be completed in a reasonable time frame, while the Abovenet network (USA) did not offer sufficient connectivity between the nodes for the connection problem to be feasible. Table 4.1 below details the size parameters of all the six networks whose topologies were available through the Rocketfuel project.

For each of the four networks used in the simulations, we consider the specific scenario where $r = 4$, $k = 1$, $\rho = 10$, $|T| = 4$, $\omega = 3 \times 10^5$ and $\omega' = 3 \times 10^3$, and show the results in the rest of this section.

We first consider the mean network costs for the different methods of information dissemination. Figures 4.1(a) to 4.1(d) show the plots of the mean network costs (μ) against p.

Table 4.1: Sizes of the six network topologies.

Network	Number of nodes ($\lvert V\rvert$)	Number of directed edges ($\lvert E\rvert$)	$\lvert E\rvert : \lvert V\rvert$ ratio	$\lvert E\rvert : \lvert V\rvert^2$ ratio
Abovenet	141	748	5.30	3.76×10^-2
Ebone	87	322	3.70	4.25×10^-2
Exodus	79	294	3.72	4.71×10^-2
Sprint	315	1944	6.17	1.96×10^-2
Telstra	108	306	2.83	2.62×10^-2
Tiscali	161	656	4.07	2.53×10^-2

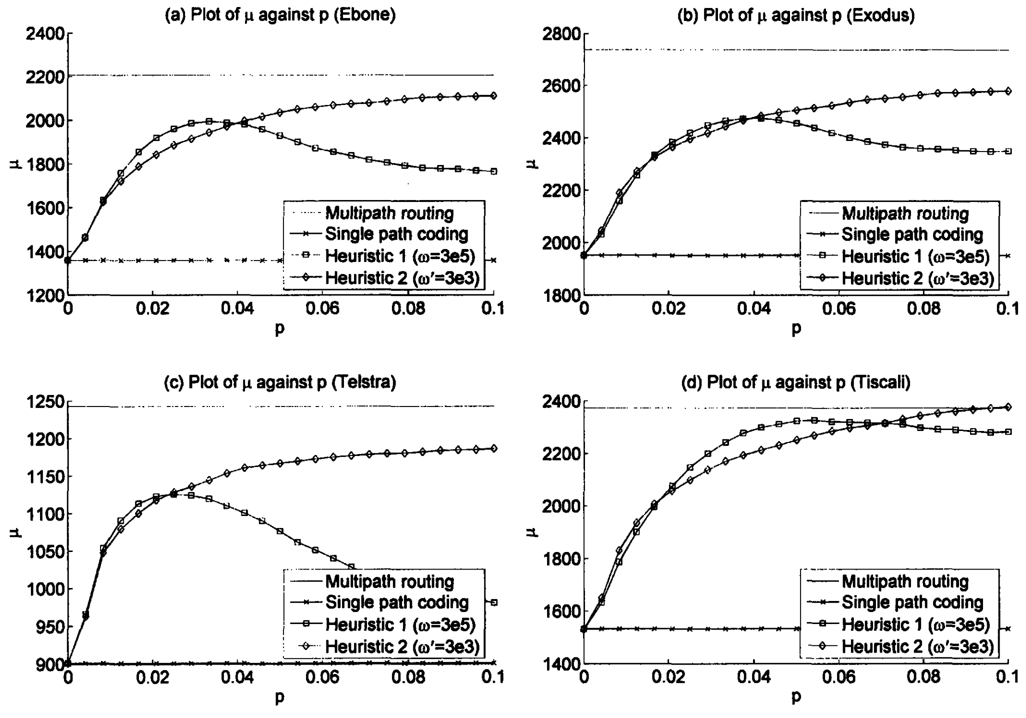

Figure 4.1: Plot of μ against p for four different networks. ($r = 4$, $\rho = 10$, $k = 1$, $|T| = 4$) (a) Ebone. (b) Exodus. (c) Telstra. (d) Tiscali.

From Figures 4.1(a) to 4.1(d), we see that the cost of single path coding is always the lowest, while the cost of multipath routing is always the highest. This is true for all four networks and for all values of p between 0 and 0.1.

For the two proposed heuristics, we first note that the mean network cost for the second heuristic is an increasing function in terms of p. This behavior is expected because as p increases, the heuristic will spread more information flow across the network from links of lower cost to links of higher cost, resulting in an increase in the network cost.

However, for the first heuristic, although the network cost first increases with increasing p when p is small, it starts to decrease or remain the same with increasing p when p is large. This seemingly odd behavior can be explained as follows.

When p is small, ν' can be lowered through the spreading of information flows. Thus, when p increases, the first heuristic attempts to spread more information flow across the network to lower ν', resulting in an increase in network cost. However, when p is large, the spreading of information flow no longer offers much benefit in the lowering of ν', and sometimes, may even result in an increase in ν' (see below for a clearer explanation of this phenomenon). Thus, as p increases, the first heuristic starts to reduce the spreading of information flow to lower network cost and to prevent a possible increase in ν', resulting in a decrease in network cost.

To further explain this queer observation, we shall provide a simple example to illustrate that the spreading of flows may not always lead to a lowering of ν'. Consider a simple unicast connection problem where five unit entropy rate input messages are to be transmitted from a source node s to a sink node t. Suppose there are five edge-disjoint paths between s and t, and each of these disjoint paths is composed of twenty unit cost edges. We employ network coding and consider two methods of information dissemination: (1) send all five messages along one of these disjoint paths, and (2)

Table 4.2: Values of ν' for the simple example to show that the spreading of flows does not always decrease ν'.

Method	ν' ($p = 0.01$)	ν' ($p = 0.1$)
(1) Single path coding	0.18	0.88
(2) Multipath coding	0.0034	0.98

send information at unit rate on each of the five disjoint paths (i.e. spread information flow across the network). For each of the above methods, we consider the value of ν', which is the probability that the wiretapper obtains at least five units of flow from the network. Table 4.2 shows the values of ν' for two different values of p.

From the table, we see that when $p = 0.01$, the spreading of flows decreases ν' from 0.18 to 0.0034, but when p increases to 0.1, the spreading of flows across the five disjoint paths increases ν' from 0.88 to 0.98 instead. This shows that the spreading of information flow across the network does not always decrease ν'.

Next, we consider the mean network vulnerabilities for the different methods of information dissemination. Figures 4.2(a) to 4.2(d) show the plots of the mean network vulnerabilities (ν) against p.

From the figures, we first note that the strategy that yields the highest network vulnerability is always the single path routing strategy, followed by the single path coding one. In addition, when p is small, the two proposed heuristics have very similar network vulnerabilities and yield lower network vulnerabilities than the multipath routing strategy. However, when p is large, the multipath routing strategy always yields the lowest network vulnerability.

When p is small, both proposed heuristics attempt to reduce their network vulnerabilities through the spreading of information flow. Thus, the similarity in the network vulnerabilities of the two heuristics is not entirely surprising. However, when p is large, the second heuristic continues to spread information flow across the network,

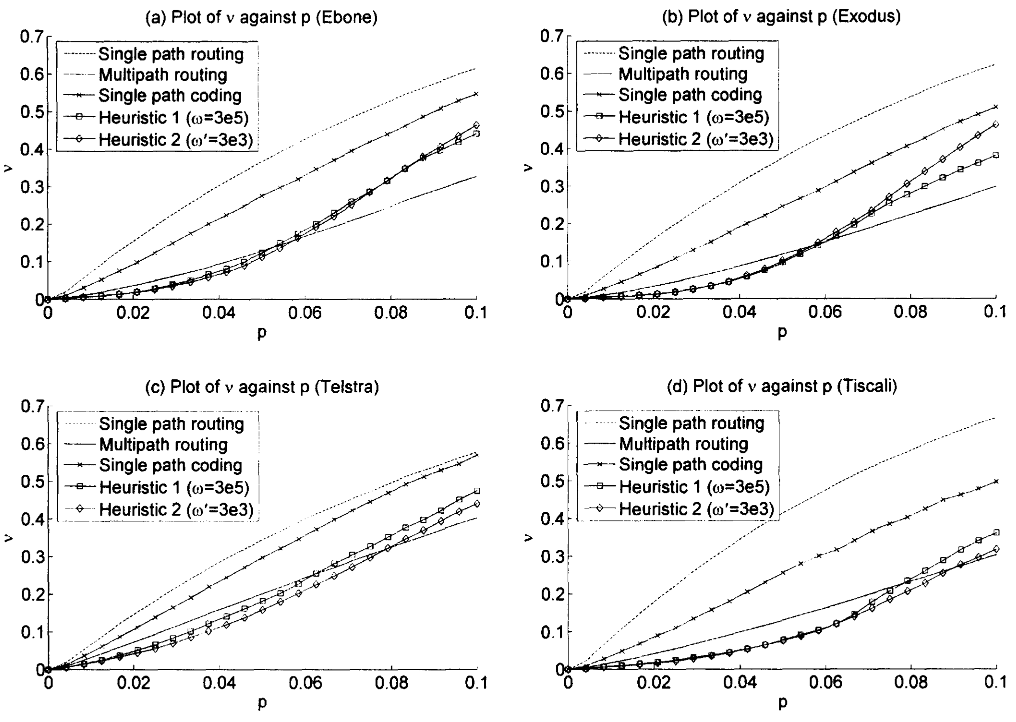

Figure 4.2: Plot of ν against p for four different networks. ($r = 4$, $\rho = 10$, $k = 1$, $|T| = 4$) (a) Ebone. (b) Exodus. (c) Telstra. (d) Tiscali.

while the first heuristic starts to limit the spread of information flow. This results in the greater deviation in their vulnerability performances as p gets larger.

To understand why the two heuristics have lower network vulnerabilities than multipath routing when p is small, but higher vulnerabilities when p is large, we return to the coded network shown in Figure 3.1(b), where the wiretapper is interested in X_1.

When p is very small, we can approximate the situation as one where the wiretapper has negligible probability of tapping two or more network links. Thus, the only way for him to retrieve X_1 is by tapping one of the three links that are carrying X_1 in an uncoded fashion. For a similar routing strategy on the network, at least four network links must be used for the multicast of X_1, and the tapping of any one of these links will enable the retrieval of X_1. Thus, routing results in a higher network vulnerability than network coding. In general, when p is small, better security is achieved by network coding as $\mathscr{Y}_{tapped}$ is often small, and the wiretapper is unable to retrieve enough degrees of freedom to decode $\mathscr{W}_{interest}$ entirely.

When p is large, the higher network vulnerability of the coding solutions can be attributed to the fact that there are potentially more ways to retrieve the messages of interest in a coded network than in a routed one. In routed networks, we first note that the tapping of links carrying messages outside $\mathscr{W}_{interest}$ does not help the wiretapper in the retrieval of the messages in $\mathscr{W}_{interest}$. However, in coded networks, the tapping of such links can potentially aid the wiretapper. Consider the network in Figure 3.1(b). Although $H(X_1 \mid X_1 + X_2) = H(X_1) > 0$ and $H(X_1 \mid X_2) = H(X_1) > 0$, we note that $H(X_1 \mid \{X_1 + X_2, X_2\}) = 0$. In general, for coded networks, there are more ways for the wiretapper to decode the messages in $\mathscr{W}_{interest}$ from $\mathscr{Y}$. This reduces the security advantage of transmitting encoded messages, and results in the higher values of ν seen in the two heuristics when p is large.

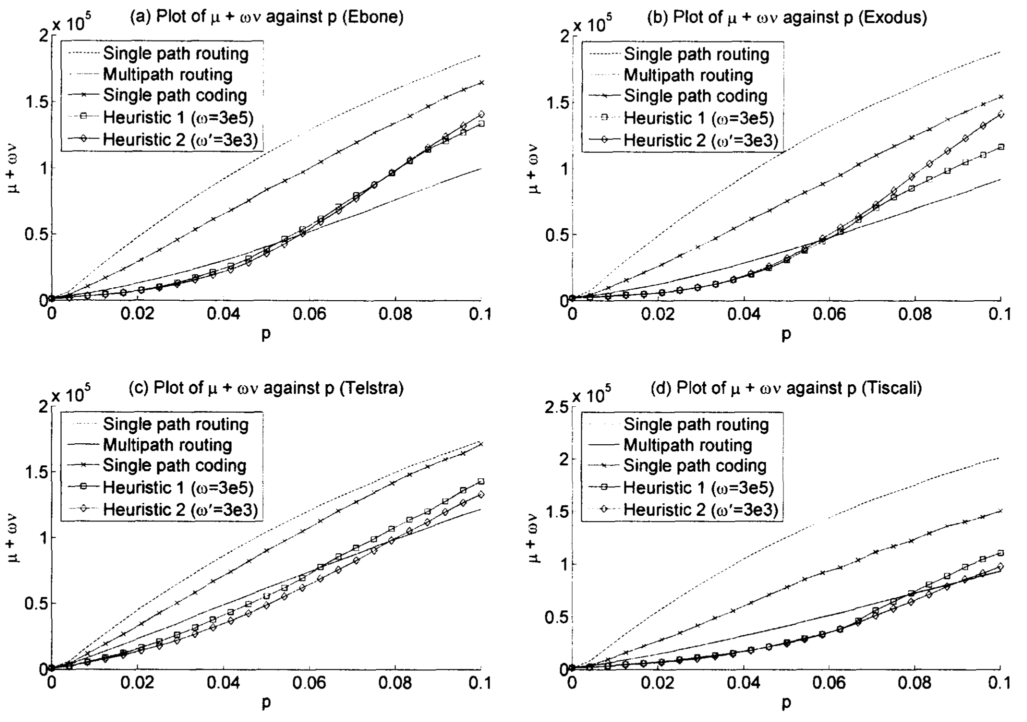

Figure 4.3: Plot of $\mathcal{F}(\mu, \nu)$ against p for four different networks. ($r = 4,\ \rho = 10,\ k = 1,\ |T| = 4$) (a) Ebone. (b) Exodus. (c) Telstra. (d) Tiscali.

Next, we consider the objective function $\mathcal{F}(\mu, \nu) = \mu + \omega\nu$. Figures 4.3(a) to 4.3(d) show the plots of $\mathcal{F}(\mu, \nu)$ against p for the four different networks.

From the figures, we note that the shapes of the graphs of Figure 4.3 are very similar to those of Figure 4.2. This is because $\omega >> \mu$ in each case, causing the influence of ν on $\mathcal{F}(\mu, \nu)$ to predominate over the influence of μ on the objective function.

Finally, we look at the difference between ν and ν' for the first heuristic for the four different networks. Figure 4.4 shows the percentage difference between ν and ν' ($\frac{\nu' - \nu}{\nu} \times 100\%$) against p.

From the figure, we note that ν' is often greater than ν, and that the absolute percentage difference between ν and ν' is usually less than 30% (with the exception of the Tiscali network).

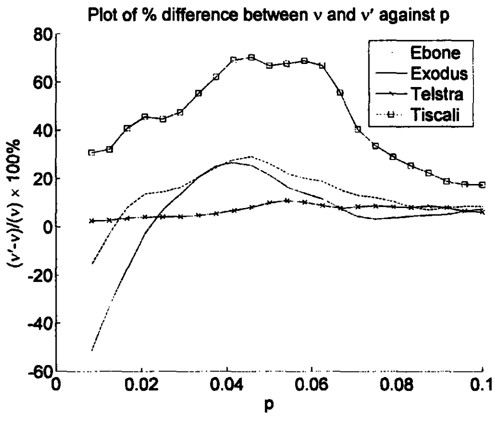

Figure 4.4: Plot of the percentage difference between ν and ν' against p for four different networks. $(r = 4,\ \rho = 10,\ k = 1,\ |T| = 4)$

Across the networks, we see that the Tiscali network returns the greatest percentage difference across the four networks, while the Telstra network usually returns the lowest. One possible explanation is that the percentage difference is highly dependent on the network size and density.

From Table 4.1, one can see that the Tiscali network has the highest number of nodes and edges, but the lowest $|E| : |V|^2$ ratio out of the four tested networks. The large size and low density of the network may have resulted in a lower probability of coding across the $r\rho$ input processes, and a lower probability that every newly tapped process will give the wiretapper an extra degree of freedom to decode the messages of interest. Thus, when compared to the other networks, the actual situation for the Tiscali network deviates more from the scenario where the first heuristic is an exact solution, resulting in the larger percentage differences observed for the Tiscali network.

On the other hand, although the Telstra network is similar to the Tiscali network in terms of density, the Telstra network is much smaller in size. This results in a lower average number of edges on the shortest path between two nodes, and a higher probability that every newly tapped process will give the wiretapper an extra degree of freedom to decode the messages of interest. Thus, when compared to the Tiscali network, the situation in the Telstra network is more similar to the scenario where the first heuristic is an exact solution, resulting in the smaller percentage differences observed for the Telstra network.

4.3 Tradeoff relationship between μ and ν

This section discusses the tradeoff relationship between network cost and network vulnerability. To investigate this relationship, we consider several scenarios and make use of the first heuristic to minimize ν' subject to a cap on the network cost μ. For each scenario, there is a minimum network cost μ_{min} for which the connection is feasible, and a maximum network cost μ_{max} for which the value of ν' is minimized. As such, when we impose the caps on the network cost, we only need to concern ourselves with cap values between μ_{min} and μ_{max}.

Figure 4.5 shows the plot of the network vulnerability ν against the network cost μ for three different values of p, with μ between μ_{min} and μ_{max}.

From the figures, we see that the graphs are all decreasing and convex, showing that the greatest decrease in vulnerability comes from the initial spreading of flows (that results in the increase in μ). In addition, when p is small, a significant decrease in ν can be obtained at the expense of having a higher network cost. In Figures 4.5(a) and (b), we see that a 90% decrease in ν can be obtained at the expense of a 30% increase in μ. However, at higher values of p, a less significant decrease in ν can be obtained. For instance, from Figure 4.5(c), we see that a 12% decrease in ν will require a 8% increase in μ.

As mentioned in the previous section, the plots of $\mathcal{F}(\mu, \nu)$ against p are greatly influenced by the vulnerability ν, because of the high value of ω used in the sim-

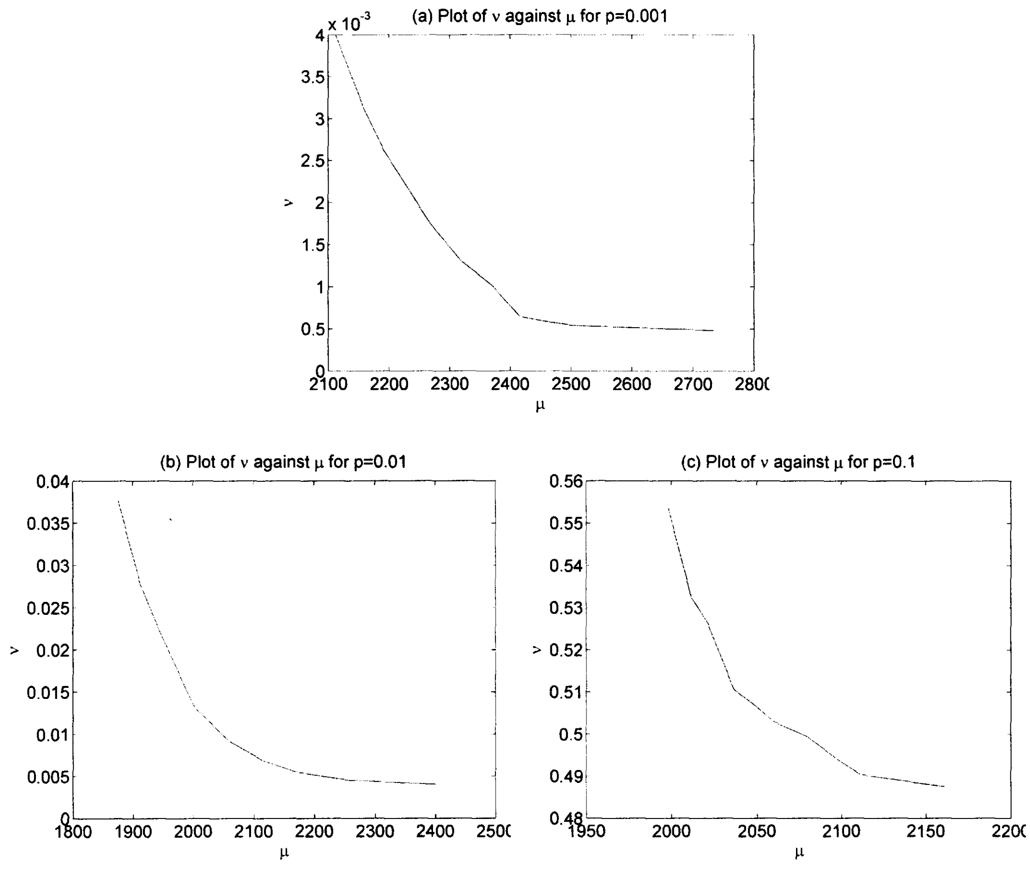

Figure 4.5: Plot of ν against μ for three different values of p. (Exodus network, with $r = 4,\ \rho = 10,\ k = 1,\ |T| = 4$)

ulations. As such, to investigate better the relationship between network cost and network vulnerability as p changes, we make use of the cost-vulnerability product ($\mu\nu$). We choose this particular function because it captures the percentage changes in both μ and ν similarly well. Although the use of this function implies the existence of a linear tradeoff relationship between μ and ν, we note that the graphs in Figure 4.5 can be reasonably well approximated by piece-wise linear functions (over two intervals of μ), and thus, the choice of this cost-vulnerability product is reasonable.

Figures 4.6(a) to 4.6(d) show the plots of the cost-vulnerability products against p for the four networks. From the figures, we first see smaller performance gaps between the single path strategies and the multipath ones, as compared to the plots of $\mathcal{F}(\mu, \nu)$ against p. This observation can be attributed to the higher network costs incurred by the multipath strategies. This observation is especially evident when p is large, where we see the proposed heuristics occasionally yielding higher values of $\mu\nu$ than the single path coding strategy. In addition, when p is large, the first heuristic tends to yield lower values of $\mu\nu$ than the second heuristic, because of the lower network cost incurred by the first heuristic, and the similar network vulnerabilities of the two heuristics.

Despite these differences between Figure 4.3 and Figure 4.6, we continue to witness better performances from the proposed heuristics than the multipath routing strategy for low values of p.

4.4 Effects of varying ω on the first heuristic

We first investigate the effects of varying the non-negative weighting variable (ω) in the objective function $\mathcal{F}(\mu, \nu)$. We note that out of the five different methods of data transmission, only the first heuristic makes use of ω to allocate flows in the network. Thus, we consider only the effects of varying values of ω on the performance of the

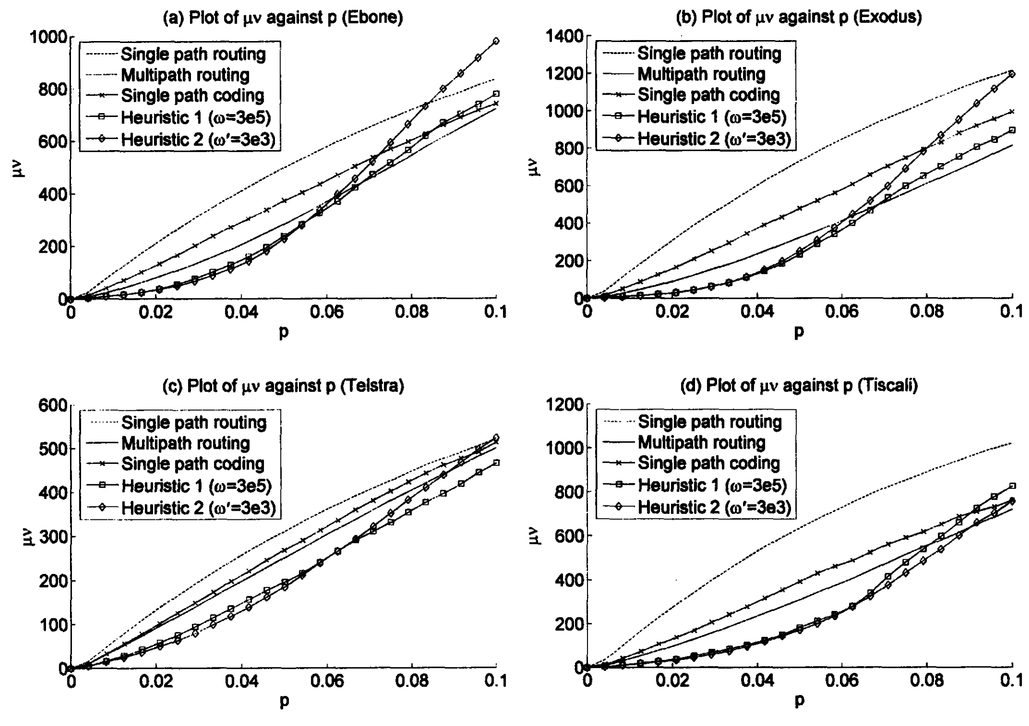

Figure 4.6: Plot of $\mu\nu$ against p for four different networks. ($r = 4$, $\rho = 10$, $k = 1$, $|T| = 4$) (a) Ebone. (b) Exodus. (c) Telstra. (d) Tiscali.

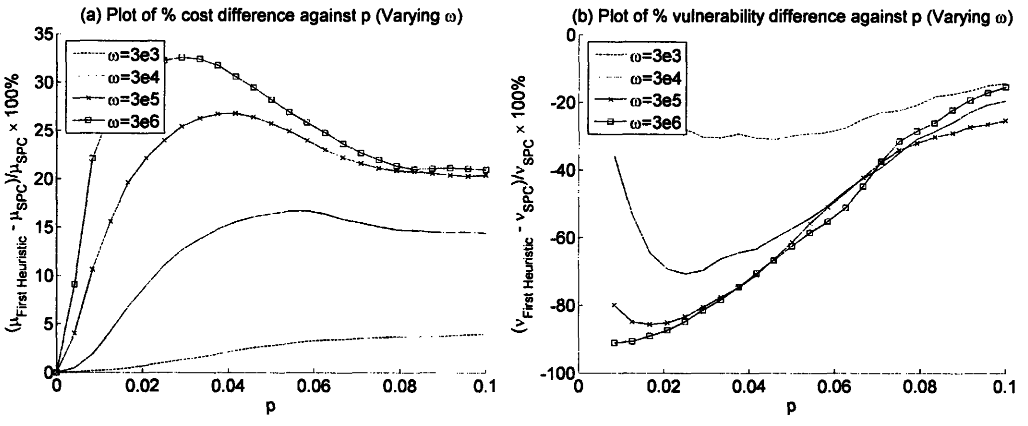

Figure 4.7: Plot of the percentage cost and vulnerability differences between the first heuristic and the single path coding strategy against p for varying values of ω. (Exodus network, with $r = 4,\ \rho = 10,\ k = 1,\ |T| = 4$)

first heuristic.

We look at the effect of ω on the network cost (μ) and the network vulnerability (ν) yielded by the first heuristic through a comparison with the corresponding values yielded by the single path coding (SPC) strategy. Figure 4.7(a) shows the plot of the percentage cost difference between the two strategies against p for different values of ω between 3×10^3 and 3×10^6, while Figure 4.7(b) shows the corresponding plot of the percentage vulnerability difference between the two strategies.

From Figure 4.7, we observe that as ω increases, the percentage cost difference increases, while the percentage vulnerability difference decreases. This observation is expected because as ω increases, more emphasis is placed on the network vulnerability than on the network cost in the objective function to be minimized. As a result, in order to minimize the function $\mu + \omega\nu'$, the first heuristic will reduce ν' at the expense of increasing μ.

Another observation is that greater changes are observed in the cost and vulnerability differences when ω increases from a small value, than when it increases from a large one. Greater changes in the two differences are also observed when p is small,

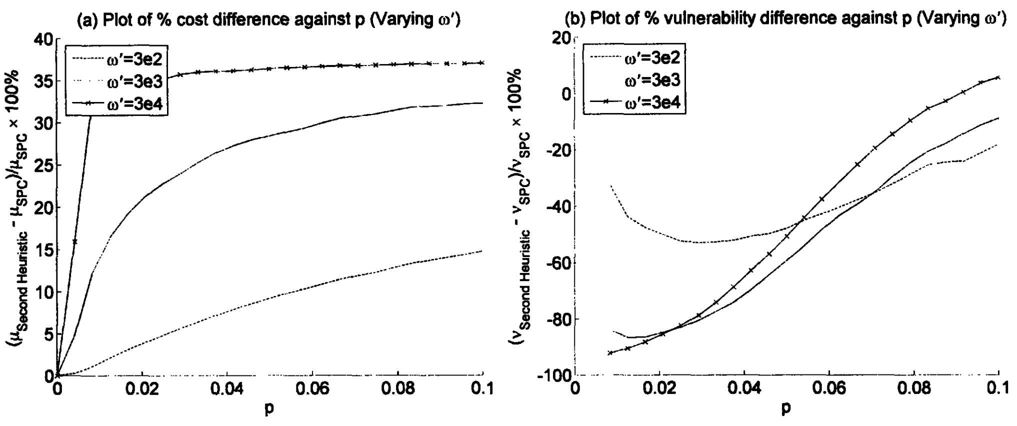

Figure 4.8: Plot of the percentage cost and vulnerability differences between the second heuristic and the single path coding strategy against p for varying values of ω'. (Exodus network, with $r = 4$, $\rho = 10$, $k = 1$, $|T| = 4$)

than when p is large. The main reason for these observations is that when ω or p is large, $\omega\nu'$ becomes much greater than μ, and the objective function $\mu + \omega\nu'$ starts to approximate $\omega\nu'$. Thus, any further increase in the value of ω will not have much effect on the way flows are allocated across the network, and the network cost does not change much.

4.5 Effects of varying ω' on the second heuristic

We go on to investigate the effects of ω' on the second heuristic. Figure 4.8(a) shows the plot of the percentage cost difference between the second heuristic and the single path coding strategy against p for different values of ω' between 3×10^2 and 3×10^4, while Figure 4.8(b) shows the corresponding plot of the percentage vulnerability difference between the two strategies.

From Figure 4.8(a), we observe that as ω' increases, the cost of the second heuristic increases. This is expected because with increasing values of ω', an increasing amount of information flow is spread across the network in order to reduce μ_{quad} at the expense of increasing μ.

Another observation is that when ω' is equal to 3×10^4, the percentage cost difference starts to taper off at 37% for increasing values of p. This is also expected because there is a fundamental limit to the amount of information flow that can be diverted from low cost paths to high cost ones.

From Figure 4.8(b), we observe that an increase in ω' does not usually lead to a decrease in the vulnerability difference (unlike what we have observed for the first heuristic in the previous section, where an increase in ω tends to decrease the vulnerability difference). In addition, from the figure, we observe that when ω' is equal to 3×10^4, the vulnerability difference actually becomes positive for values of p larger than 0.09, indicating that the second heuristic starts to yield a higher network vulnerability than the single path coding strategy.

The explanation for this observation is that when p is small, the spreading of flows reduces the probability that the adversary is able to tap a large number of messages from the network. Thus, as ω' increases, the size of the set $\mathscr{Y}_{tapped}$ generally decreases, making it harder for the wiretapper to decode the messages in $\mathscr{W}_{interest}$. This causes the vulnerability difference to become increasingly negative as ω' increases.

However, when p is large, an increase in the spreading of information flow no longer significantly decreases the probability that the adversary is able to tap a large number of messages from the network. (Recall in Section 4.2, when p is large, the spreading of flows no longer decreases ν' significantly, resulting in the decreasing network cost of the first heuristic as p increases.) Instead, a second consequence of the spreading of flows starts to take effect, whereby more opportunities for coding across the messages are created. This results in an increase in the probability that every newly tapped process will give the wiretapper an additional degree of freedom to decode the messages in $\mathscr{W}_{interest}$. Thus, the greater spreading of flows, resulting

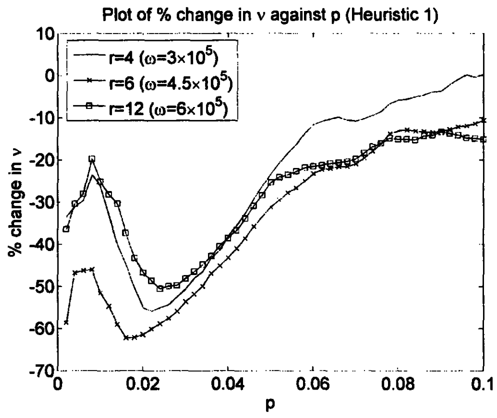

Figure 4.9: Plot of the percentage change of ν against p as r increases from 2. (Exodus network, $\rho = 10,\ k = 1,\ |T| = 4$.)

from the increase in ω', starts to increase the network vulnerability when p is large.

4.6 Effects of varying r

As mentioned earlier, although we do not explicitly consider the possibility of increasing network security at the expense of increased network cost by introducing additional messages into the network, we can still investigate the effects of the introduction of these additional messages by keeping k constant while increasing the value of r.

Figure 4.9 shows the plot of the percentage change in ν for the first heuristic (as compared to the reference case where $r = 2$, $\omega = 1.5 \times 10^5$), as the value of r varies. The values of ω were appropriately scaled to compensate for the higher values of μ with increasing r.

From the figure, we observe that, when r increases from 2 to 6, ν decreases, but when r increases further to 12, ν does not decrease anymore. On the contrary, ν increases for small values of p when $r = 12$. This interesting behavior may be explained as follows.

When r increases, it affects the network vulnerability in several ways. First, the mean number of encoded processes X_i in the transmitted processes $Y(l')$ grows as $O(r)$. This decreases ν as the wiretapper will, on average, need to retrieve more degrees of freedom from the network in order to decode all the messages in $\mathscr{W}_{interest}$. Second, the size of $\mathscr{Y}$ grows as $O(2^r)$, increasing the number of ways $\mathscr{W}_{interest}$ can be decoded from the messages in $\mathscr{Y}$, thus increasing ν. Finally, the average amount of flow on the network links grows as $O(r)$. Thus, for any fixed p, the expected size of $\mathscr{Y}_{tapped}$ increases, and ν increases.

As the increase in r can affect the network vulnerability in these different ways, the net effect of an increase in r on ν is ambiguous. However, the simulation results suggest that when r is small, an increase in r decreases ν, while for larger values of r, an increase in r increases ν. This is probably due to the different growths of the network parameters mentioned in the previous paragraph. As the size of $\mathscr{Y}$ grows as $O(2^r)$, when r is small, the growth of $\mathscr{Y}$ is small, and the net effect of an increase in r is a decrease in ν. However, when r gets large, the growth of $\mathscr{Y}$ becomes much larger, resulting in a net increase in ν.

4.7 Concluding remarks

From the results obtained through the simulations, we see that when the network links are not too susceptible to being tapped (i.e. p is small), our proposed heuristics can provide lower values of network cost and network vulnerability than multipath routing. However, when p is large, our heuristics yield lower costs, but higher vulnerabilities than multipath routing. Despite the poorer vulnerability performance, it is important to note that the primary aim of this research work is to keep both the network cost and the network vulnerability low. Thus, there may be times where the poorer vulnerability performance by the heuristics may be more than outweighed by their better cost performance.

Our results have also shown that the heuristics are rather robust to changes in network parameters such as the network sizes, the number of input messages, and the non-negative weighting variables ω and ω'. In particular, we note that the first heuristic is often more robust than the second heuristic to changes in the network parameters.

Finally, we will like to point out several ways in which the performances of the heuristics can be improved or be made more robust. First, the vulnerability performance of the heuristics can be improved through changes that decrease the number of ways the messages in $\mathscr{W}_{interest}$ can be decoded from $\mathscr{Y}$. These changes include the reduction of the size of the finite field over which the coding is to be done, and the use of deterministic codes, although these changes often bring with them other undesirable consequences. For instance, a reduction in the finite field size inevitably reduces the probability that the messages in $\mathscr{W}$ can be successfully decoded at the sink nodes.

Second, as both our proposed heuristics are currently unaffected by changes in the values of k, $|V|$ and $|E|$, improvements can be made to them to improve their performances in different situations. Taking the first heuristic for instance, instead of defining ν' to be the probability that at least $r\rho$ network processes are tapped, it can be defined as the probability that at least $r\rho f(k)$ processes are tapped, where $f(k)$ is an increasing and possibly concave function of k.

Chapter 5

Conclusion and future work

In this thesis, we have considered the problem of providing a set of connections at both a low cost and a high level of security against wiretapping. As an exact solution is difficult, we have presented two heuristic methods of finding the coding subgraphs that can achieve this. Through our simulation results, we observe that a trade-off between network cost and network vulnerability also exists in coded networks. In addition, we observe that network coding can be more effective than traditional routing for low cost and secure data multicast, especially when the links are not too easily tapped.

In this study, we have focused on the problem of finding the coding subgraph, and conducted our simulations in a centralized manner. However, it should be noted that the finding of the coding subgraph can done in a decentralized manner for the second heuristic that we have proposed, since it has a strictly convex cost function for each network link [6]. As we have assumed the use of a random linear network code, which itself is implementable in a distributed manner, we conclude that our second heuristic solution to the problem can be implemented in an entirely decentralized manner.

As we have only considered the issue of resilience against wiretapping in this work, future research can be done to investigate the other security issues of network coding. Furthermore, as we have only suggested two simple heuristic approaches to the problem, alternate algorithmic approaches remain as clear avenues for future work.

Bibliography

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

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

[3] T. Ho, M. Médard, J. Shi, M. Effros and D. R. Karger, "On randomized network coding," *Proc. - 41st Annual Allerton Conference on Communication Control and Computing*, Oct. 2003.

[4] R. Koetter, and M. Médard, "Beyond routing: An algebraic approach to network coding," *Proc. - IEEE INFOCOM*, vol. 1, pp. 122 - 130, 2002.

[5] D. S. Lun, M. Médard, T. Ho, and R. Koetter, "Network coding with a cost criterion," *IEEE International Symposium on Information Theory and its Applications*, 2004.

[6] D. S. Lun, N. Ratnakar, R. Koetter, M. Médard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," *Proc. - IEEE INFOCOM*, Mar. 2005.

[7] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *Proc. 41st Annual Allerton Conference on Communication, Control, and Computing*, Oct. 2003.

[8] N. Cai and R. W. Yeung, "Secure network coding," *Proc. - IEEE International Symposium on Information Theory*, pp. 323, 2002.

[9] J. Feldman, T. Malkin, R. A. Servedio and C. Stein, "On the capacity of secure network coding," *Proc. - 42nd Annual Allerton Conference on Communication, Control and Computing*, September 2004.

[10] T. Ho, B. Leong, R. Koetter, M. Médard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," *Proc. - IEEE International Symposium on Information Theory*, pp. 144, 2004.

[11] J. Jeong, and J. Ma, "Security analysis of multi-path routing scheme in ad hoc networks,"

Lecture Notes in Computer Science, vol. 3320, pp. 694 - 697, 2004.

[12] W. Lou, W. Liu, and Y. Fang, "SPREAD: Enhancing data confidentiality in mobile ad hoc networks," *Proc. - IEEE INFOCOM*, vol. 4, pp. 2404 - 2413, 2004.

[13] J. Yang, and S. Papavassiliou, "Improving network security by multipath traffic dispersion," *Proc. - IEEE MILCOM*, vol. 1, pp. 34 - 38, 2001.

[14] C. K.-L. Lee, X.-H. Lin, and Y.-K. Kwok, "A multipath ad-hoc routing approach to combat wireless link insecurity," *IEEE International Conference on Communications*, vol. 1, pp. 448 - 452, 2003.

[15] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," *IEEE/ACM Trans. Networking*, vol. 12, no. 1, pp. 2 - 16, Feb. 2004.

[16] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379 - 423, 623 - 656, Jul. and Oct. 1948.