# Paper Review-MAP Detection of Misbehaving Relay

Harshvardhan Singh
*Electronics and Comm. Engg.*
*Shiv Nadar University*
Greater Noida, India
hs504@snu.edu.in

Moesha Malik
*Electronics and Comm. Engineering*
*Shiv Nadar University*
Greater Noida, India
mm396@snu.edu.in

Garneni Jhansi
*Electronics and Comm. Engineering*
*Shiv Nadar University*
Greater Noida, India

***Abstract*—MAP Detection of Misbehaving Relay in Wireless Multiple Access Relay Networks utilises or Math in Paper Title or Abstracts.**

***Index Terms*—Multiple Access Relay Networks, MAP Detection, Misbehaving Relay, bit error, detection**

## I. Introduction

Network coding is a new relaying technique that replaces the traditional store and forward paradigm of network routing by a method that allows intermediate (relay) nodes to mix the received data before re-transmission.

This has been shown to maximize throughput, as well as robustness. While network coding can be an efficient means of information dissemination in networks, it also presents a new security challenge as a single corrupted packet has the potential to corrupt every packet received by a given destination.

The problem of detecting misbehaving relays that inject false data in single-source networks has been studied in. The receiver then computes the ground truth of the tracing bits and compares them with the tracing bits received from a relay to determine whether it is malicious or cooperative. All these major previous works, however, require sending extra reference data (overhead) at the source to detect the misbehaving relay.

## II. Previous Works

The problem of detecting misbehaving relays that inject false data in single-source networks has been studied in the papers referred below:

### *A. Signatures for Content Distribution*

In *"Signatures for Content Distribution with Network Coding, in Proc. IEEE International Symposium on Information Theory, June 2007"* [1] the authors consider a peer-to-peer (P2P) network in which the source generates a signature vector and broadcasts to all nodes where it is used to check the integrity of the received packets. The paper proposes a new signature scheme that is based on and is designed specifically for random linear coded systems. The signature scheme makes use of the linearity property of random linear network coding, and enables the peers to check the integrity of packets without the requirement for a secure channel, as in the case of hash function or Secure Random Checksum (SRC) schemes. Also, the computation involved in the signature generation and verification processes is very simple.

### *B. Byzantine Modification Detection in Multicast Networks with Random Network Coding*

There are several Information Theory based algorithms proposed for mitigating Byzantine modification attack. In *"Byzantine modification detection in multicast networks with random network coding, IEEE Trans. Inf. Theory, vol. 54, no. 6, pp. 2798-2803, June 2008"* and *"Resilient network coding in the presence of Byzantine adversaries, IEEE Trans. Inf. Theory, vol. 54, no. 6, pp. 2596-2603, June 2008.)"*. Byzantine modification detection capability can be added to a multicast scheme based on random linear block network coding, with modest additional computational and communication overhead, by incorporating a simple polynomial hash/check value in each packet.

### *C. Miscellaneous Methods*

In *"Tracing malicious relays in cooperative wireless communications, IEEE Trans. Inf. Forensics Security, vol. 2, no. 2, pp. 198-212, June 2007"* the authors consider inserting tracing bits in the data stream at the source in a cryptographically secure manner. The receiver then computes the ground truth of the tracing bits and compares them with the tracing bits received from a relay to determine whether it is malicious or cooperative.

Extensions to multiple-source networks have been studied in *"Error analysis in multi-source, multi-relay, multi-destination networks under falsified data injection attacks," in Proc. IEEE MILCOM, 2008* and *"An algebraic watchdog for wireless network coding, in Proc. IEEE International Symposium on Information Theory, June 2009)*, where the tracing bits or polynomial hash functions are used in detecting the misbehaving relays.

## III. Map Detection Scheme

The paper proposes the maximum a posteriori (MAP) approach in **detecting the misbehaving relay that may inject false data**. The MAP detection scheme is **based on the log-likelihood ratio (LLR) test which is optimal** in the sense of

minimizing the probability of incorrect decisions (false alarm and misdetection).

### A. Advantages of Scheme

The given scheme has the following relative advantages to the previously described error detection schemes, which included additional storage (to observe patterns for tampering) and/or transmission overheads (to add parity checks etc.) hence adding to overages which can have significant impact while transmitting large amounts of data.

- Does not require sending extra bits at the source, such as hash function or message authentication check bits, and hence there is **no transmission overhead**
- Makes an instantaneous decision about whether a relay is behaving properly without a long term observation.
- Derives the probability of **false alarm and mis-detection as a function of the signal-to-noise ratio (SNR)**, taking into account the lossy nature of wireless links.
- Side information regarding the presence of relay misbehavior is exploited at the destination (decoder) to mitigate the relay misbehavior and enhance the reliability of decoding.
- The MAP decoding scheme exploits the likelihood of the presence of relay misbehavior, and show that it provides a dramatic improvement over the conventional decoder that does not exploit the side information.

### B. System Model Description

The system taken under consideration, takes two sources, one relay and one destination (as shown in Fig 1).We're considering a MA Relay Network, where the source and destination cannot communicate to each other directly because the distance between the source and destination is greater than the transmission range of both of them, hence the need for intermediate node(s) to relay. The relay "overhears" the bits and forwards the coded bit ($p = x_1^r \oplus x_2^r \oplus f$) to the destination.

$$p = x_1 \oplus x_2 \oplus z \tag{1}$$

$$where, z = e_1 \oplus e_2 \oplus f$$

We assume the signals to be sent through orthogonal Rayleigh fading channels with additive white Gaussian noise and path loss

- Giving us the expression for the received signals at destination as $y_i = h_i * x_i \sqrt{d_i^{-m} * E_s} + n_i, i = 1, 2$ as received signals from $i^{th}$ source and $y_r = h_r p \sqrt{d_r^{-m} E_r} + n_r$ for the received signal from the relay
- m is path loss exponent
- Between the $i^{th}$ source and destination, and relay and destination:
  - $h_i$ and $h_r$ is channel fading gain (Gaussian variables with mean zero and variance one)
  - $d_i$ and $d_r$ is distance between source and destination, and relay and destination respectively
  - $E_s$ and $E_r$ is transmitted energy per symbol
  - $n_i$ and $n_r$is noise at destination (mean zero and variance $N_o/2$ per dimension)

### C. Detection Rule

The destination works to find whether $z$ is +1 (well-behaving) or -1 (mis-behaving) to point out if the relay is misbehaving or not. To find out the LLR of $z$ we break it into its constituent component $(p \oplus x_i)$ to find out
LLR of $x_i$ after knowing $h_i$ and $y_i$

$$L_i = ln \frac{P(x_i = +1|h_i, y_i)}{P(x_i = -1|h_i, y_i)} = \frac{4\sqrt{d_i^m}}{N_o} Re[h_i^* y_i]$$

and LLR of $p$, after knowing $h_r$ and $y_r$

$$L_r = ln \frac{P(x_r = +1|h_r, y_r)}{P(x_r = -1|h_r, y_r)} = \frac{4\sqrt{d_r^m}}{N_o} Re[h_r^* y_r]$$

thus deriving our LLR of $z$ as

$$L(z|h, y) = ln \frac{P(z = +1|h, y)}{P(z = -1|h, y)} \approx sign(L_r * L_1 * L_2).min[|L_r||L_1||L_2|] \tag{2}$$

The value of Eqn. 2 will be used to determine the **estimation** of the value of $z$ as:

$$\hat{z} = +1, \quad if L(z|h, y) \geq 0$$

$$\hat{z} = -1, \quad if L(z|h, y) < 0$$

```
# Set parameters for the model
d_i = 1  # Distance b/w source and relay
E_s = 1  # Normalised SNR measure - Source
d_r = 1  # Distance between relay and reciever
E_r = 1  # Normalised SNR measure - Relay
m = 0.7  # Path loss exponent
N_o = 0.25  # Variance

def z_estimation_calculate(h, y):
    L_r = ((4 * pow(pow(d_r, -m) * E_r, 0.5)) / N_o) * (np.conj(h[0]) * y[0]).real
    L_1 = ((4 * pow(pow(d_i, -m) * E_s, 0.5)) / N_o) * (np.conj(h[1]) * y[1]).real
    L_2 = ((4 * pow(pow(d_i, -m) * E_s, 0.5)) / N_o) * (np.conj(h[2]) * y[2]).real

    if L_r * L_1 * L_2 < 0:
        L_z = -1 * min(abs(L_r), abs(L_1), abs(L_2))
    elif L_r * L_1 * L_2 >= 0:
        L_z = min(abs(L_r), abs(L_1), abs(L_2))
    print(f"L_r: {L_r}\nL_1: {L_1}\nL_2: {L_2}")
    print(f"LLR value of z is: {L_z}")
    return L_z
```

## IV. Decoding Schemes

The paper also places under consideration, three decoding schemes, depending on the side information regarding $z$:

### A. MAP Decoder

The MAP decoder calculates the LLR of $x_1$(or $x_2$) given $h$ and $y$, $L(x_1|h,y)$, and decide $\hat{x}_1 = +1$ if $L(x_1|h,y) > 0$ and $\hat{x}_1 = -1$ otherwise.
It considers $(x1, x2, pt \oplus \hat{z})$ as a valid code-word, where $p_t = x1 \oplus x2$ is the true parity bit, and finds the most probable (closest) code-word given a received vector (MAP decoding).

- Therefore, if a false alarm ($\hat{z}$ = -1 given $z$ = +1) occurs, then the decoder considers (x1, x2, -pt) as a valid code-word while (x1, x2, pt) is valid.
- Similarly, if a miss detection ($\hat{z}$ = 1 given $z$ = -1) occurs, then the decoder considers (x1, x2, pt) as a valid code-word while (x1, x2, -pt) is valid.

In case a wrong parity bit is applied, the reliability of decoding will be decreased.

```
def test_z_estimator():
    print("Enter '+1' for binary 1 and '-1' for binary 0")
    x1 = int(input("Enter source 1 output: "))
    x2 = int(input("Enter source 2 output: "))
    p = int(input("Enter relay output: "))
    predict = z_estimation(x1, x2, p)
    if predict >= 0:
        print("Decoded output is +1")
        print(f"Valid code-word: ({x1}, {x2}, +1)")
    elif predict < 0:
        print("Decoded output is -1")
        print(f"Valid code-word: ({x1}, {x2}, -1)")
    else:
        print("Incorrect operation")
```

### B. Genie Decoder

The Genie-aided decoder assumes the availability of perfect side information regarding z, i.e. $\hat{z} = z$, hence $P_{FA} = P_{MD} = 0$. Therefore, the decoder considers (x1, x2, -pt) as a valid code-word when code-words are generated as (x1, x2,-pt) and, similarly, (x1, x2, pt) as a valid code-word when code-words are generated as (x1, x2, pt). This corresponds to the case of fully cooperative relay (and remains purely theoretical), and serves as a reference for performance comparison with other decoders.

### C. Conventional Decoder

Conventional decoder does not have the side information regarding $z$. Hence, the decoder considers (x1, x2, pt) as a valid code-word no matter what $z$ is. Therefore, if $z$ = -1, the conventional decoder considers (x1, x2, pt) as a valid code-word while (x1, x2,pt) is valid. Similarly, if $z$ = 1, it considers (x1, x2, pt) as a valid code-word while (x1, x2, pt) is valid. This corresponds to a special case of the proposed decoder with $P_{FA} = 0$ and $P_{MD} = 1$.

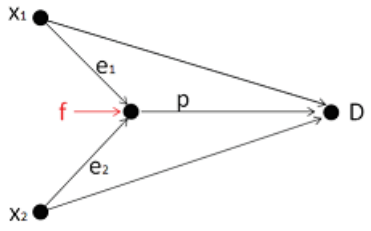


Fig 2. Decoder reference relay diagram

## V. Numerical and Graphical Results

### A. False Alarm and Mis-detection

The above described system is what is described as a soft-detect system, the paper computes a theoretical probability of false alarm $P_{FA}$ and mis-detection $P_{MD}$ v/s the fraction of energy allotted to each source $\alpha = \gamma_s/\gamma_b$ (where $\gamma_s$ is SNR of each source, and $\gamma_b$ is received SNR per information bit, given by $\gamma_b = (2\gamma_s + \gamma_r)/2$

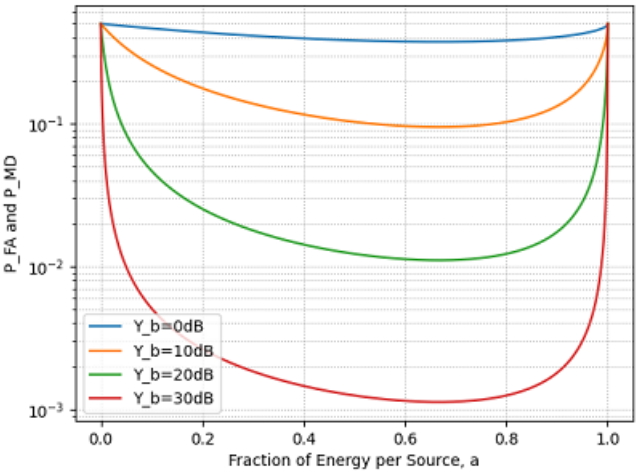


Fig 3. Probability of false alarm and mis-detection v/s fraction of energy per source

```
def P_Graph():
    for SNR_b in [1,10,100,1000]:
        x=[]
        y=[]
        for i in range(0,100001):
            a=i/100000
            x.append([a])
            SNR_r = 2*(1-a)*SNR_b
            SNR_s = a*SNR_b
            fac = pow(SNR_r/(1+SNR_r),0.5)* (SNR_s/(1+SNR_s))
            P_fa = 0.5*(1-fac)
            y.append([P_fa])
        plt.plot(x,y,label=f"Y_b={int(round(10*math.log(SNR_b,10),0))}dB")
    plt.xlabel('Fraction of Energy per Source, a')
    plt.ylabel('P_FA and P_MD')
    plt.yscale('log')
```

```
    plt.legend(loc="lower left")
    plt.grid(True, which="both",ls="dotted")
    plt.show()
```

### B. Probability of Bit-Error

Let $\hat{x}_1$ denote the decoder output for $x_1$ at the destination. The probability of bit error for $x_1$ can be shown to be [2] (Proof at Appendix A)

*a) MAP Decoder:*

$$\begin{aligned} P_{E,1} &= P(\hat{x}_1 \neq x_1|\mathbf{h,y}) \\ &= \frac{(1-P_{MD})P(z=-1)+(1-P_{FA})P(z=+1)}{1+\exp(|L_1+L(p\oplus x_2|h_r,y_r,h_2,y_2)|)} \\ &+ \frac{P_{FA}P(z=+1)+P_{MD}P(z=-1)}{1+\exp(|L_1-L(p\oplus x_2|h_r,y_r,h_2,y_2)|)} \end{aligned} \tag{3}$$

where

$$L(p\oplus x_2 h_r,y_r,h_2,y_2) = \ln\frac{\exp(L_r)+\exp(L_2)}{1+\exp(L_r+L_2)}$$

and

$$L(p\oplus x_2\oplus(-1)|h_r,y_r,h_2,y_2) = -L(p\oplus x_2|h_r,y_r,h_2,y_2) \tag{4}$$

and a similar method can be used to derived bit error for $x_2$

- For the case of genie decoder, we can modify Eq. 3 to accommodate for $P_{MD}$=0 and $P_{FA}$=0, to give:]

$$\begin{aligned} P_{E,1} &= P(\hat{x}_1 \neq x_1|\mathbf{h,y}) \\ &= \frac{1}{1+\exp(|L_1+L(p\oplus x_2|h_r,y_r,h_2,y_2)|)} \end{aligned} \tag{5}$$

- And for a conventional decoder, which is a special case of a genie decoder, with $P_{FA}$=0 and $P_{MD}$ =1, and it follows from (3) and (4), that we get

$$\begin{aligned} P_{E,1} &= P(\hat{x}_1 \neq x_1|\mathbf{h,y}) \\ &= \frac{P(z=+1)}{1+\exp(|L_1+L(p\oplus x_2|h_r,y_r,h_2,y_2)|)} \\ &+ \frac{P(z=-1)}{1+\exp(|L_1-L(p\oplus x_2|h_r,y_r,h_2,y_2)|)} \end{aligned} \tag{6}$$

```
def BER_estimation_calculate(h, y, SNR_b, a,↩
     no):
    SNR_r = 2 * SNR_b * (1 - a)
    SNR_s = a * SNR_b

    L_r = round(((4 * pow(SNR_r, 0.5)) / pow↩
        (N_o,0.5)) * abs((np.conj(h[0]) * y↩
        [0]).real), 2)
    L_1 = round(((4 * pow(SNR_s, 0.5)) / pow↩
        (N_o,0.5)) * abs((np.conj(h[1]) * y↩
        [1]).real), 2)
    L_2 = round(((4 * pow(SNR_s, 0.5)) / pow↩
        (N_o,0.5)) * abs((np.conj(h[2]) * y↩
        [2]).real), 2)
    print(L_r,L_1,L_2)
    P_fa = 0.5 * (1 - (pow(SNR_r / (1 + ↩
        SNR_r), 0.5) * (SNR_s / (1 + SNR_s)))↩
        )
    P_md = P_fa
    j = math.log((math.exp(L_r) + math.exp(↩
        L_2)) / 1 + math.exp(L_r + L_2))  # ↩
        to find BER for x1
    pf, p1, p2 = 1, 1e-2, 1e-2
    Pz_1 = pf * (1 + 2*p1 + 2*p2 + 3*p1*p2) ↩
        + (p1 + p2 + 2 * p1*p2)
    Pz1 = 1 - Pz_1
    if no == 1:
        ans = 1 / (1 + math.exp(abs(L_1 + j)↩
            ))
        return ans
    elif no == 2:
        ans = (((1 - P_md) * Pz_1 + (1 - ↩
            P_fa) * Pz1) / (1 + math.exp(abs(↩
            L_1 + j)))) + ((P_md * Pz_1 + ↩
            P_fa * Pz1) / (1 + math.exp(abs(↩
            L_1 - j))))
        return ans
    elif no == 3:
        ans = (Pz1 / (1 + math.exp(abs(L_1 +↩
             j)))) + (Pz_1 / (1 + math.exp(↩
            abs(L_1 - j))))
        return ans
```

The above BER models can then be plotted for particular values of $\alpha$ (2/3 in this case, for lowest $P_{FA}$ and $P_{MD}$ ) and can be printed using:

```
def BER_Graph():  # To print Graph1
    x = []
    y1 = []
    y2 = []
    y3 = []
    a = 2 / 3
    for i in range(0, 161):
        x_no = i/10
        SNR_b = x_no #pow(10, x_no*10)
        x.append([x_no])
        P_E1 = BER_estimation(SNR_b, a, 1)
        y1.append([P_E1])
        P_E1 = BER_estimation(SNR_b, a, 2)
        y2.append([P_E1])
        P_E1 = BER_estimation(SNR_b, a, 3)
        y3.append([P_E1])

    plt.plot(x, y1, label="Conventional")
    plt.plot(x, y2, label="Proposed")
    plt.plot(x, y3, label="Genie-aided")
    plt.xlabel('Received SNR per information↩
         bit (in dB)')
    plt.ylabel('Probability of Bit Error (↩
        P_E1)')
    plt.yscale('log')
    plt.legend(loc="lower left")
    plt.grid(True, which="both", ls="dotted"↩
        )
    plt.show()

def BER_estimation(SNR_b, a, no):
    x1, x2, p = -1, -1, -1
    h_r = complex(1, 0.5)
```

```
    h_1 = complex(1, 0.5)
    h_2 = complex(1, 0.5)
    y_r = complex(y_calculate(h_r, E_r, p))
    y_1 = complex(y_calculate(h_1, E_s, x1))
    y_2 = complex(y_calculate(h_2, E_s, x2))
    h = [h_r, h_1, h_2]
    y = [y_r, y_1, y_2]
    return BER_estimation_calculate(h, y, ↩
        SNR_b, a, no)
```

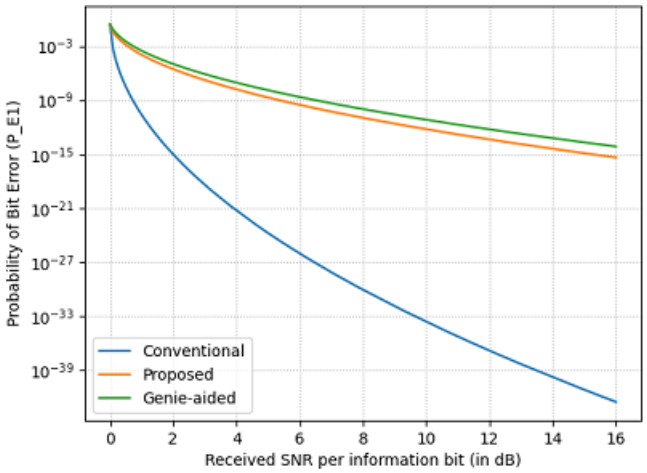


Fig 4. Probability of Bit-error v/s Recieved SNR per information bit

## VI. Scope for further extensions to paper

Based on the above results, we can extend the scope of the paper to recommend the following extensions to the paper

*a) Reducing number of Sources::* It can be intuitively deduced that with lower number of sources, we face lower chances of the relay admitting a false bit, and consequently lower rate of error

*b) Adding Tracing bits:* The process of adding tracing bits involves adding reference bits who's locations are known at the destination. Furthermore, the literature on LLR concludes that for a bit with absolute certainty, the magnitude of LLR for known bit is infinity, thus increasing the total LLR magnitude of z.

*c) Adopting better soft-detect methods::* While the LLR detection method has been well-known for being used for soft-detection methods, we can further try and derive additional information about the transmitted data, and based on the kind of data being transmitted factor it in during the decoding stage.

## References


[1] F. Zhao, T. Kalker, M. Medard, and K. J. Han, "Signatures for Content Distribution with Network Coding," IEEE International Symposium on Information Theory, 2007.

[2] S. W. Kim, "Mitigation of Forwarding Misbehavior in Multiple Access Networks with Network Coding," IEEE Global Telecommunications Conference GLOBE-COM 2010, 2010.

[3] Yinian Mao and Min Wu, "Tracing Malicious Relays in Cooperative Wireless Communications" 2007 IEEE Transactions on Information Forensics and Security.