Lattice Codes in Wireless Networks

Su Li

Autumn 2013

Contents

1 Introduction

The work presented in this report is realized in semester project, during autumn 2013. In this project I try to use a modified compute-and-forward scheme[1] which utilizes the Channel State Information (CSI) at the transmitters. By using varied version of the same lattice at the transmitter and the receiver, it is possible to make profit on the CSI to better balance the rates of different users and get a larger rate region.

This report is oranised as following. First we give a breif expanation about the modified version compute and forward. Then we use it to some channel model, such as downlink distributed antenna system, multiple access channel and many-to-one channel.

2 Compute and Forward with CSI

An new observation[2] shows that the lattice with respect to which the decoder performs the lattice decoding does not have to be the same lattice in which the transmitted codeword lies.

Consider a relay network with K transmitters and M relays. The discrete-time real Gaussian channel can be represented as following

$$y_m = \sum_{k=1}^{K} h_{mk} x_k + z_m, \quad m \in [1 : M] \tag{1}$$

with$y_m \in \mathbb{R}^n$, $x_k \in \mathbb{R}^n$, $h_{mk} \in \mathbb{R}$ denoting the channel output of relay m, channel input of transmitter k and channel gain, respectively. $z_m \in \mathbb{R}^n$ is Gaussian white noise with unit variance for each component.

When codeword t_k is given to encoder, the channel input is forming as following

$$x_k = \left[\frac{t_k}{\beta_k} + d_k\right] mod \Lambda_k^s / \beta_k \tag{2}$$

where d_k is dither and uniformly distributed in voronoi region ν_k^s/β_k. Thans to the dither, x_k is independent from t_k and also uniformly in ν_k^s/β_k hence has average power P for all k.

As pointed out in[3] , rate tuple $(R_1, \ldots, R_K)$ is achievable if

$$\begin{aligned} R_k &< r_k\left(h_m, a_m, \beta_{1:K}\right) \\ &:= \max_{\alpha_m \in \mathbb{R}} \frac{1}{2} \log^+ \left(\frac{\beta_k P}{\alpha_m^2 + P \left\|\alpha_m h_m - \widetilde{a}_m\right\|^2}\right) \\ &= \frac{1}{2} \log^+ \left(\left\|\widetilde{a}_m\right\|^2 - \frac{P\left(h_m^T \widetilde{a}_m\right)^2}{1 + P\left\|h_m\right\|^2}\right)^{-1} + \frac{1}{2} \log \beta_k^2 \end{aligned} \tag{3}$$

for all k with $\widetilde{a}_m := [\beta_1 a_{m1}, \ldots, \beta_K a_{mK}]$. Without loss of generality, we can set any of β_k to be 1.

Comparing to the fomula of standard compute and forward, it is easy to find that we can set different β_k to each user according to their channel state. As a result, those users of better channels do not have to compromise with the worst one.

3 Reverse Compute and Forward

3.1 Problem statement

Taking advantages of the linear structure of the lattice codes and the additive nature of Gaussian interference networks, the compute and forward scheme is a novel coding schemel for Gaussian networks. The basic idea is to decode linear combiantion of messages rather than single message itself at the receiver. Such idear works well when the receiver has many different observations of the

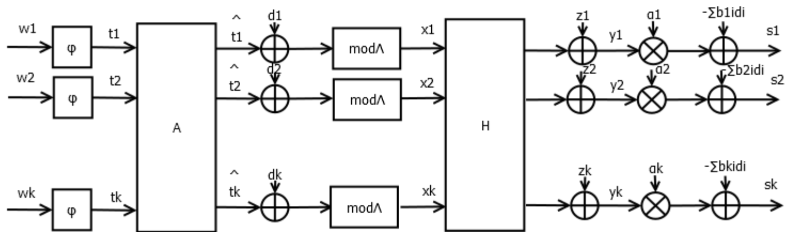

Figure 1: System Model for Reverse Compute and Forward

message. However, in some cases, for example downlink distributed antenna system, each receiver have only one antenna and they locate in different place which means no cooperation among them.

In such situation, if any of those receivers wants to get their desired message, it has to decode at least two equaiton and at most as many as the user number equations, which depends one the channel coefficients. Here we do not consider the case that receiver only has to decode one equation and get the best rates, since it is too special and it treat messages of other users as noise, which dose not take the advantages of the linear structure of the lattice codes.

We also know that the achievable rate is monotone nonincreasing as the number of equations gets larger. So the primary idea is decoding as less number of equation as possible but still can get the demanding message. This idear is proposed in [2] which mainly focus on the channel with discrete addictive noise. But we find that it still works when the noise is continous.

3.2 System model and Scheme

Consider the downlink distributed antenna system as figure1, where the transmitter has K messages, each of which is only needed in the corresponding receiver.

When messages tuple $(t_1, \ldots, t_K)$ are given to encoder, first encoder makes new tuple $\left(\hat{t}_1, \ldots, \widehat{t}_k\right)$ as the linear combiantion of the old tuple.

$$\widehat{t}_k = \sum_{i=1}^{K} a_{ki} t_i \tag{4}$$

Then the new message tuple forms the channel input as following

$$x_k = [\widehat{t}_k + d_k] mod \Lambda \tag{5}$$

where d_k is dither which is a random vector uniformly distributed in Voronoi region, which make x_k is independent from $\widehat{t}_k$ and also uniformly distributed in Λ.

Theorem 1: For any $\varepsilon > 0$ and n large enough, there exist nested lattice codes $\Lambda \subseteq \Lambda_1 \subseteq \cdots \subseteq \Lambda_k$ with rates $R_1, \ldots, R_K$, such that for channel vector

$h_1, \ldots, h_K \in \mathbb{R}^K$ and coefficient vector $b_1, \ldots, b_K \in \mathbb{Z}^K$. User m can decode lattice

$$v_m = [t_m] mod\Lambda \tag{6}$$

with average probability of error ε so long as

$$R_k = max \frac{1}{2} \log^+ \left(\frac{P}{\alpha_m^2 + P \left\| \alpha_m h_m - b_m \right\|^2} \right) \tag{7}$$

and $B = [b_1, b_2, \ldots, b_k]$ is invertible.

Proof: The channel output at receiver m is

$$y_m = \sum_{i=1}^{K} h_{mi} x_i + z_m \tag{8}$$

and receiver computes

$$s_m = \alpha_m y_m - \sum_{i=1}^{K} b_{mi} d_i \tag{9}$$

$$\begin{aligned} [s_m]\, mod\Lambda &= \left[\sum_{i=1}^{K} (\alpha_m h_{mi} x_i - b_{mi} d_i) + \alpha_m z_m \right] mod\Lambda \\ &= \left[\sum_{i=1}^{K} (b_{mi} (x_i - d_i) + (\alpha_m h_{mi} - b_{mi}) x_i) + \alpha_m z_m \right] mod\Lambda \\ &= \left[\sum_{i=1}^{K} b_{mi} \widehat{t}_i + \sum_{i=1}^{K} (\alpha_m h_{mi} - b_{mi}) x_i + \alpha_m z_m \right] mod\Lambda \end{aligned} \tag{10}$$

To get an extimation of lattice eqaution of v_m.

$$\begin{aligned} \widehat{v}_m &= \left[\sum_{i=1}^{K} b_{mi} \widehat{t}_i \right] mod\Lambda \\ &= \left[\sum_{i=1}^{K} \sum_{k=1}^{K} b_{mi} a_{ik} t_i \right] mod\Lambda \\ &= [t_m]\, mod\Lambda \end{aligned} \tag{11}$$

If we choose $A = [a_1, a_2, \ldots, a_K] = B^{-1}$.

When B is not invertible, we can still choose A to make it happen for some of the receivers that they only have to decode one equation to get the desired messages. For those who cannot enjoy such advantage just need to decode more equations to get their messages as what they do in standard compute and forward.

4 Compute-and-forward on MAC

4.1 Problem statement

In order to achieve the capacity on non-corner point of multiple access channel, we usually use time sharing. However, it is shown that the compute-and-forward is able to achiveve those points on the boundary of the capacity region of a two-user MAC[4]. It is not known that whether such property still occurs when the number of user increases.

4.2 Three User MAC

We first consider the three-user MAC

$$y = h_1 x_1 + h_2 x_2 + h_3 x_3 + Z \tag{12}$$

with equal constraints P. Without loss of generality we can set $P = 1$ and $\beta_1 = 1$.

There are two trivial cases which can achieve the capacity boundary. One is decode identity matrix as equation matrix which give the corner points on the capacity boundary. Another is using the first equation to decode one message and treat others as noises. In such way, we can reduce the dimention to 2, which is known that we can achieve the capacity bounds by using compute and forward. So we focus on when the first equation cannot give any message.

4.2.1 One Capacity Achievable Equation Matrix

Theorem 2: Decoding the following eqaution matrix

$$\begin{bmatrix} a & b & c \end{bmatrix} = \begin{bmatrix} a_1 & 1 & 0 \\ a_2 & 0 & 1 \\ a_3 & 0 & 0 \end{bmatrix} \tag{13}$$

can achieve the capacity boundary of 3-user-MAC, if the following inequalities are satisfied.

$$\begin{gathered} \beta_2^2 \beta_3^2 \left(1 + \|h\|^2\right)^2 \geq \left[\left(1 + \|h\|^2\right) \|\widetilde{a}\|^2 - \left(h^T \widetilde{a}\right)^2\right]^3 \\ \beta_2^4 \beta_3^4 \left(1 + \|h\|^2\right) \geq \left[\left(1 + \|h'\|^2\right) \|\widetilde{a}'\|^2 - \left(h'^T \widetilde{a}'\right)^2\right]^3 \\ a_3 \leq \beta_2 \end{gathered} \tag{14}$$

where $\widetilde{a} = [a_1\beta_1, a_2\beta_2, a_3\beta_3]^T$, $h' = [h_2, h_3]^T$ and $\widetilde{a}' = [a_2\beta_2, a_3\beta_3]^T$.

proof: Decode the first equation $a = [a_1, a_2, a_3]^T$, we have the rates

$$R_1(a) = \frac{1}{2} \log^+ \frac{1}{\widetilde{a}^T \left(I + hh^T\right)^{-1} \widetilde{a}} \tag{15}$$

$$R_1(a) = \frac{1}{2} \log^+ \frac{\beta_2}{\widetilde{a}^T \left(I + hh^T\right)^{-1} \widetilde{a}} \tag{16}$$

$$R_1(a) = \frac{1}{2}\log^+ \frac{\beta_3}{\widetilde{a}^T\left(I + hh^T\right)^{-1}\widetilde{a}} \tag{17}$$

If we could achieve the boundary of the capacity region, we should have

$$R_1(a) + R_2(a) + R_3(a) \geq C_{sum} = \frac{1}{2}\log\left(1 + \|h\|^2\right) \tag{18}$$

which is equivalent to

$$\beta_2^2\beta_3^2\left(1 + \|h\|^2\right)^2 \geq \left[\left(1 + \|h\|^2\right)\|a\|^2 - \left(h^T a\right)^2\right]^3 \tag{19}$$

Using SIC we decode the second equation $b = [b_1, b_2, b_3]^T$ with the rates

$$R_1(b \mid a) = \frac{1}{2}\log \widetilde{b}^T\left[I - \frac{\widetilde{a}\widetilde{a}^T}{\|a\|^2} - \frac{\left(I - \frac{\widetilde{a}\widetilde{a}^T}{\|a\|^2}\right)hh^T\left(I - \frac{\widetilde{a}\widetilde{a}^T}{\|a\|^2}\right)}{1 + h^T\left(I - \frac{\widetilde{a}\widetilde{a}^T}{\|a\|^2}\right)h}\right]\widetilde{b} \tag{20}$$

$$R_2(b \mid a) = R_1(b \mid a) + \frac{1}{2}\log\beta_2^2 \tag{21}$$

$$R_3(b \mid a) = R_1(b \mid a) + \frac{1}{2}\log\beta_3^2 \tag{22}$$

where $\widetilde{b} = [b_1\beta_1, b_2\beta_2, b_3\beta_3]^T$.

By choosing $b = [1, 0, 0]^T$, we can simplify the results and get

$$R_1(b \mid a) = \frac{1}{2}\log\frac{\left(1 + \|h\|^2\right)\|a\|^2 - \left(h^T a\right)^2}{\left(1 + \|h'\|^2\right)\|a'\|^2 - (h'^T a')^2} \tag{23}$$

$$R_2(b \mid a) = \frac{1}{2}\log\frac{\left(\left(1 + \|h\|^2\right)\|a\|^2 - \left(h^T a\right)^2\right)\beta_2^2}{\left(1 + \|h'\|^2\right)\|a'\|^2 - (h'^T a')^2} \tag{24}$$

$$R_3(b \mid a) = \frac{1}{2}\log\frac{\left(\left(1 + \|h\|^2\right)\|a\|^2 - \left(h^T a\right)^2\right)\beta_3^2}{\left(1 + \|h'\|^2\right)\|a'\|^2 - (h'^T a')^2} \tag{25}$$

If we could achieve the boundary of the capacity region, we should have

$$R_1(b \mid a) + R_2(b \mid a) + R_3(b \mid a) \geq C_{sum} = \frac{1}{2}\log\left(1 + \|h\|^2\right) \tag{26}$$

which is equivalent to

$$\beta_2^2\beta_3^2\left[\left(1 + \|h\|^2\right)\|a\|^2 - \left(h^T a\right)^2\right]^3 \geq \left[\left(1 + \|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]^3\left(1 + \|h\|^2\right) \tag{27}$$

together with (19) we can get

$$\beta_2^4\beta_3^4\left(1 + \|h\|^2\right) \geq \left[\left(1 + \|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]^3 \tag{28}$$

Using SIC we only have to decode $c = [0,1,0]^T$ or $c = [0,0,1]^T$. Suppose we choose the former one, then we can get rates

$$R_1\left(c \mid a, b\right) = \frac{1}{2}\log\left(\left[\left(1+\|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]\right) - \frac{1}{2}\log\left(a_3^2\beta_3^2\right) \quad (29)$$

$$R_2\left(c \mid a, b\right) = \frac{1}{2}\log\left(\left[\left(1+\|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]\right) - \frac{1}{2}\log\left(\frac{a_3^2\beta_3^2}{\beta_2^2}\right) \quad (30)$$

$$R_{13}\left(c \mid a, b\right) = \frac{1}{2}\log\left(\left[\left(1+\|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]\right) - \frac{1}{2}\log\left(a_3^2\right) \quad (31)$$

then we can have sum rate

$$R_{sum}\left(c \mid a, b\right) = \frac{1}{2}\log\left(\left[\left(1+\|h'\|^2\right)\|a'\|^2 - \left(h'^T a'\right)^2\right]\right)^3 - \frac{1}{2}\log\left(\frac{a_3}{\beta_2}\right)^6 \quad (32)$$

If equation (28) holds, then we have

$$R_{sum}\left(c \mid a, b\right) = \frac{1}{2}\log\left(1+\|h\|^2\right) - \frac{1}{2}\log\left(\frac{a_3}{\beta_2}\right)^6 \quad (33)$$

which means if $a_3 \leqq \beta_2$, then we can achieve the capacity, since the rates we got from previous two eqautions both achieve the capacity.

4.2.2 Possible Case

Form previous results we can conclude that if the first and second equaiton can solve at least one message out, then we can choose β to achieve the capacity boundary. The result is not clear if one message can be sloved from the first and second equations. However we belive that under some certain condition, it is possible to achieve capacity for this case.

4.2.3 Numeric Example

In the simulation, we set power $P = 5$ and channel gain coefficient $H = [1, 1.5, 2]$. In figure 2, the blue points on the boundary are obtained by the two trivial cases we mentioned above. The red points are cases that we decode $a = [0,1,1]$as the first equation. As can be seen from the figure, two red points is overlaped with two blue points. According to the symmetry, it seems that there should exits more red points, but it may be because the parameters we choose is too small which make them cannot happen. The green points are cases that we decode $a = [1,1,1]$as the first equation. Results of all red and green points satisfy the three conditions as we pointed at (14).

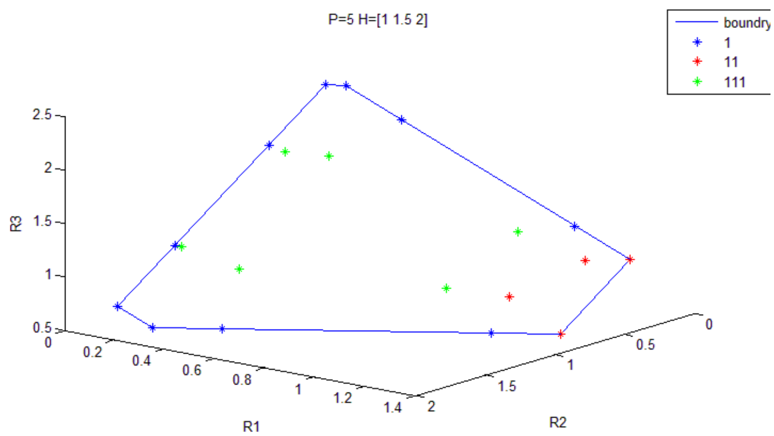

Figure 2: Example of COF on 3 User MAC

4.3 N User MAC

Consider the general case, when the number of user is N, the following equation matrix is capacity achievable

$$\begin{bmatrix} a_1 & 1 & 0 & \cdots & 0 \\ a_2 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & 1 \\ a_N & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} a & \begin{matrix} I \\ 0 \end{matrix} \end{bmatrix} \tag{34}$$

where $a_N \neq 0$. The provement is similar to what we have done in 4.2.1. After decoding the first and second equations, we have successfully reduce the dimension to $N-1$. Using induction we can prove that this equation matrix is capacity achievable.

5 COF on Many-to-one Interference Channel

During the summer, we have explored the relationship between the number of required eqautions and the channel coefficients. We have obtained some patterns about the region for number of required eqautions. Now we want to use compute and forward with channel state information to see whether it has strong influence about the result. we run several simulations for the different power setting for the same channel $H = [1, 3.5, h_2, h_3]$ and the channel coefficients of direct links of cogitive users are 1 . The results of standard compute and forward is shown as figure 3, while the results of compute and foward with channel state information is shown as figure 4.

Give power constriant $P = 5$, in the standard compute and forward case, the number of required equations is $\begin{bmatrix} 2 & 3 & 4 \end{bmatrix}$and largest region part is for decoding 4 equations. However, with channel state information, the number of

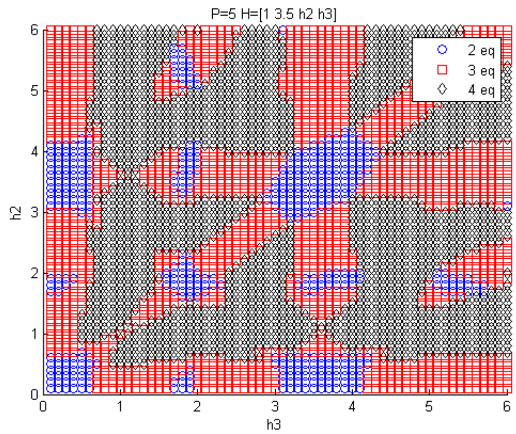

Figure 3: Simulation results for COF on Many-to-one

required equation is $\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$ and the largest region part is for decoding 2 equations. Also, if we set differents powe constraint for the modified compute and forward case, we can find that when P is small, there are still large part of the region where we need to decode 4 equations, which means decoding all the messages, not a good news. As P goes large, the region for decoding 2 equations become lagrer and for 3 and 4 equaitons become smaller. When P large enough, region for 1 equation occurs and region for 4 equations disappears.

6 Conclusion

In this semester project, I mainly study the computer-and-forward with channel state information and try to use it in some channel model and compare the performance with the standard compute-and-forward in many-to-one channel. The scheme of reverse compute and forward seems quite good when we have a invertible equaiton matrix, but how to choose proper preprocessing matrix A when B is not invertible is still a uncover part. For the compute-and-forward on MAC, we found a capacity achievable way to choose the equations. But whether the condition becomes looser or tigthter as N goes large is unkown. If I have more time, they are quite interesting things to discover.

References

[1] Bobak Nazer, Michael Gastpar: Compute-and-Forward: Harnessing Interference Through Structured Codes. IEEE Transactions on Information Theory 57(10): 6463-6486 (2011)

[2] Jingge Zhu: Documentation about using CSI in construction lattice code.

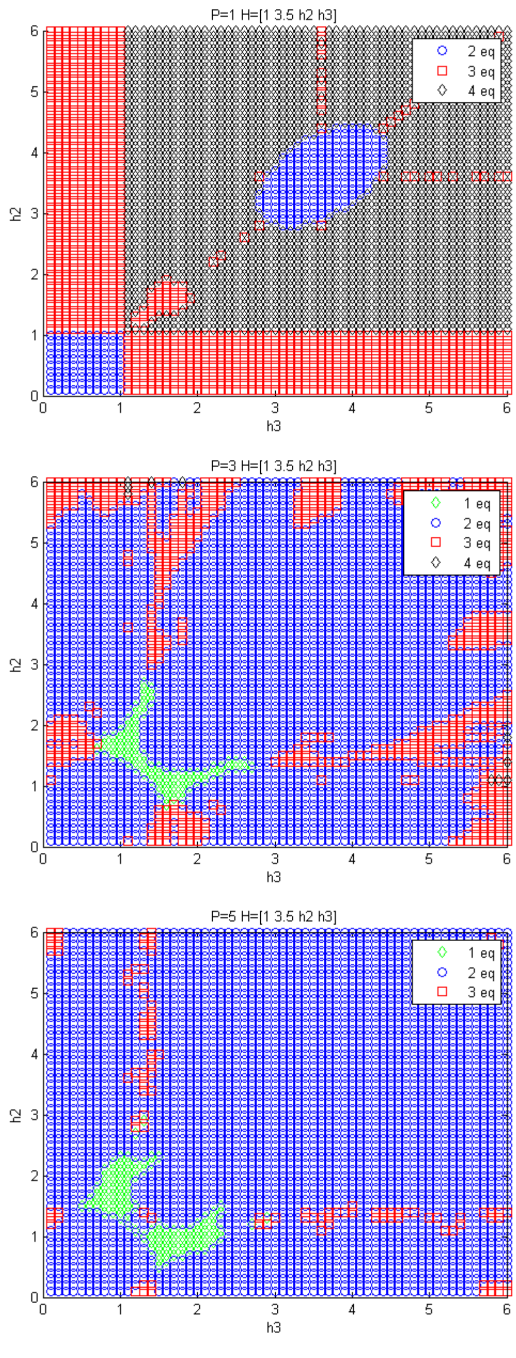

Figure 4: Simulation results for COF with SIC on Many-to-one

[3] Song-Nam Hong, Giuseppe Caire: Reverse Compute and Forward: A Low-Complexity Architecture for Downlink Distributed Antenna Systems. IEEE International Symposium on Information Theory Proceedings(2012)

[4] Jingge Zhu: Comput-and-forward on the two-user MAC.