Fifty-second Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
October 1 - 3, 2014

Two-way Function Computation

Seiyun Shin
Department of Electrical Engineering
KAIST, Daejeon, South Korea
Email: seiyun.shin@kaist.ac.kr

Changho Suh
Department of Electrical Engineering
KAIST, Daejeon, South Korea
Email: chsuh@kaist.ac.kr

***Abstract*— We explore the role of feedback for the problem of reliable computation over two-way multicast networks. Specifically we consider a scenario in which there are forward-message computation demands and feedback is offered through the backward network for aiding the forward-message computation. We characterize the *feedback computation capacity* of a four-node Avestimehr-Diggavi-Tse deterministic network in which two nodes in one side wish to compute modulo-2 sums of two independent Bernoulli sources generated from the other two nodes. As a consequence of this result, we show that the backward network can be more efficiently used for feedback, rather than if it were used for independent backward-message computation. Our achievability proof builds upon a network decomposition framework developed in our earlier work.**

I. Introduction

Earlier results on the role of feedback in communications were somewhat discouraging. In the 50s, Shannon proved that feedback has no bearing on capacity for memoryless point-to-point channels [1]. Subsequent work showed that feedback provides gain for point-to-point channels with memory [2], [3] as well as for many multi-user channels [4]–[6]. For many scenarios, capacity improvements due to feedback are rather modest. However, one notable recent result in [7] has changed the traditional viewpoint on the role of feedback. It is shown in [7] that feedback provides more significant capacity gains for the Gaussian interference channel. Subsequent work in [8] shows more promise on the use of feedback, demonstrating that feedback can provide a *net* increase in capacity even if feedback cost is taken into consideration.

Our interest is to examine the benefit of feedback for more general scenarios in which nodes now intend to compute *functions* of the raw messages rather than the messages themselves. For an idealistic perfect feedback scenario, Suh-Gastpar [9] have recently shown that feedback provides a significant gain for computation as in classical communication settings [7]. However the result does not take into account feedback cost. Whether or not there exists a feedback gain for computation in the presence of feedback cost has been unexplored. This motivates us to investigate the feedback gain for a more realistic scenario that respects feedback cost.

Specifically, we explore a computation scenario in which there are forward function-multicast traffics and the backward network is employed only for the purpose of feedback to help forward-message computation. As in [9], we consider the Avestimehr-Diggavi-Tse (ADT) deterministic network model [10] which well abstracts wireless Gaussian networks. In the context of classical communication, it has been well known that ADT networks can approximate wireless Gaussian networks within a constant gap to the optimality in capacity [10], [11]. Recently, a similar approximation result has been established for the problem of computation in which a single receiver wishes to compute a linear function of multiple Gaussian sources [12]. Specifically [12] employs lattice codes to show that a multiple source single-destination Gaussian network can be approximated to a class of linear deterministic networks (which includes the ADT network as a special case), within a constant factor of the optimal performance w.r.t. the distortion for computing the sum of Gaussian sources. We expect that this approximation approach can be applied to our computation scenario. So as an intermediate model towards the Gaussian model, we take the ADT model. Specifically we consider a four-node ADT deterministic network where the two nodes in one side want to compute modulo-2 sums of two independent Bernoulli sources generated from the other two nodes.

Motivated by the setting in [8], in order to count feedback cost, we introduce a design parameter that captures possibly different symbol rates between forward and backward networks. Specifically we define the parameter λ as the total time spent in the backward network for the purpose of feedback normalized by the total time spent in the forward network.

For this model, we develop a new achievable scheme and derive matching upper bounds, thereby establishing the *feedback computation capacity*. Our achievable scheme builds upon a *network decomposition framework* developed in [13]. As in [13], [14], we observe that for our problem setting, coding separately over decomposed orthogonal components achieves the optimal performance, i.e., the decomposition holds without loss of optimality. Moreover, from this result, we demonstrate that as in classical communication settings, the backward network can be more efficiently used for the purpose of feedback, rather than if it were used for independent backward-message computation. The gain comes from the fact that feedback enables us to exploit side information at nodes, thus making the backward network effectively more capable.

II. Model

We consider a four-node ADT deterministic network illustrated in Fig. 1. Node k $(k = 1, 2)$ sends its own message S_k^K during N time slots where S_1^K and S_2^K are assumed

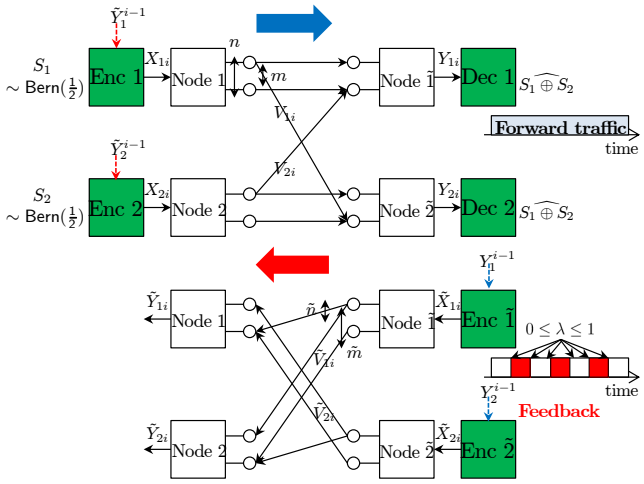

Fig. 1. A four-node ADT deterministic network.

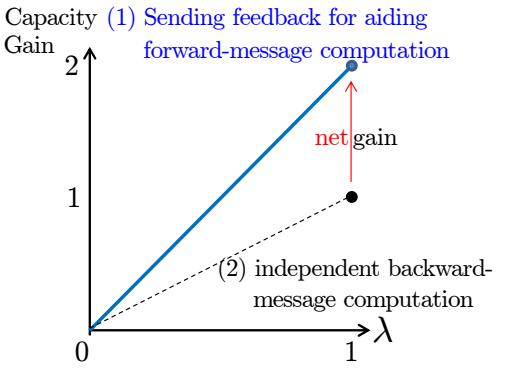

Fig. 2. An efficient use of backward networks: $(m,n)=(0,3)$, $(\tilde{m},\tilde{n})=(2,1)$.

to be independent and identically distributed according to $\mathsf{Bern}(\frac{1}{2})$. Node $\tilde{k}$ $(\tilde{k}=1,2)$ wishes to compute modulo-2 sums of the two Bernoulli sources S_1^K and S_2^K. Here we use shorthand notation to indicate the sequence up to K, e.g., $S_1^K := (S_{11},\cdots,S_{1K})$. This network consists of the forward and backward parts. In the forward network, n and m indicate the number of signal bit levels for direct and cross links respectively. The corresponding values for the backward network are denoted by $(\tilde{n},\tilde{m})$.

Let $X_k \in \mathbb{F}_2^q$ $(k=1,2)$ be an encoded signal of node k where $q=\max(m,n)$ and $V_k \in \mathbb{F}_2^m$ be part of X_k visible to node j $(\neq k)$. Similarly, let $\tilde{X}_k \in \mathbb{F}_2^{\tilde{q}}$ be an encoded signal of node $\tilde{k}$ where $\tilde{q}=\max(\tilde{m},\tilde{n})$ and $\tilde{V}_k$ be part of $\tilde{X}_k$ visible to node j $(\neq \tilde{k})$. The received signals at node k and node $\tilde{k}$ are then given by

$$Y_1 = \mathbf{G}^{q-n}X_1 \oplus \mathbf{G}^{q-m}X_2,\ \tilde{Y}_1 = \tilde{\mathbf{G}}^{\tilde{q}-\tilde{n}}\tilde{X}_1 \oplus \tilde{\mathbf{G}}^{\tilde{q}-\tilde{m}}\tilde{X}_2,$$
$$Y_2 = \mathbf{G}^{q-m}X_1 \oplus \mathbf{G}^{q-n}X_2,\ \tilde{Y}_2 = \tilde{\mathbf{G}}^{\tilde{q}-\tilde{m}}\tilde{X}_1 \oplus \tilde{\mathbf{G}}^{\tilde{q}-\tilde{n}}\tilde{X}_2,$$

where $\mathbf{G}$ and $\tilde{\mathbf{G}}$ are shift matrices and operations are performed in $\mathbb{F}_2$: $[\mathbf{G}]_{ij} = \mathbf{1}\{i=j+1\}$ $(1\le i,j\le q)$, $[\tilde{\mathbf{G}}]_{ij} = \mathbf{1}\{i=j+1\}$ $(1\le i,j\le \tilde{q})$.

The encoded signal X_{ki} of node k at time i is a function of its own message and past feedback signals: $X_{ki} = f_{ki}(S_1^K, \tilde{Y}_k^{i-1})$. We define $\tilde{Y}_k^{i-1} := \{\tilde{Y}_{kt}\}_{t=1}^{i-1}$ where $\tilde{Y}_{kt}$ denotes the feedback signal received at node k at time t. The encoded signal of node $\tilde{k}$ at time i, denoted by $\tilde{X}_{ki}$, is a function of its past received sequences Y_k^{i-1}: $\tilde{X}_{ki} = \tilde{f}_{ki}(Y_k^{i-1})$.

We consider a full-duplex system in which we can send signals through forward and backward networks simultaneously. We introduce a design parameter λ which indicates the total time spent in the backward network for the purpose of feedback normalized by the total time spent in the forward network. Note that the forward network is fully utilized for transmission. Hence $0\le\lambda\le 1$. Here $\tilde{X}_k^N$ is regarded as a whole vector that includes feedback signals as well as null signals, e.g., $\tilde{X}_k^N = \{\varnothing, \tilde{X}_{k2}, \varnothing, \tilde{X}_{k4}, \cdots\}$, thus $\sum_{i=1}^N H(\tilde{X}_{ki}) \le N\lambda\max(\tilde{m},\tilde{n})$.

Node $\tilde{k}$ uses a decoding function d_k to estimate $\{S_{1i}\oplus S_{2i}\}_{i=1}^K$ from its received signal Y_k^N. An error occurs whenever $S_{1i}\oplus S_{2i} \ne \widehat{S_{1i}\oplus S_{2i}}$ for some i. The probabilities of error are then given by $\lambda_k = P\left\{d_k \ne \{S_{1i}\oplus S_{2i}\}_{i=1}^K\right\}$, $k=1,2$. We say that the computation rate $R=\frac{K}{N}$ is achievable if there exists a family of codebooks and encoder/decoder functions such that the error probabilities of both λ_1 and λ_2 go to zero as code length N tends to infinity. We define the computation capacity C as the supremum of all achievable computation rates.

III. Main Results

Theorem 1 (Feedback computation capacity): Let $\alpha := \frac{m}{n}$.

$$C = \begin{cases} \min\{C_{\mathsf{no}}+\lambda\tilde{m}, C_{\mathsf{pf}}\}, & \alpha<1,\\ \min\{C_{\mathsf{no}}+\lambda\tilde{n}, C_{\mathsf{pf}}\}, & \alpha>1,\\ C_{\mathsf{no}}, & \alpha=1.\end{cases}$$

where C_{no} and C_{pf} indicate the nonfeedback and perfect feedback computation capacities respectively [13], [9]:

$$C_{\mathsf{no}} = \begin{cases} \min\{m,\frac{2}{3}n\}, & \alpha<1,\\ \min\{n,\frac{2}{3}m\}, & \alpha>1,\\ n, & \alpha=1.\end{cases}$$

$$C_{\mathsf{pf}} = \begin{cases} \frac{2}{3}n, & \alpha<1,\\ \frac{2}{3}m, & \alpha>1,\\ n, & \alpha=1.\end{cases}$$

Proof: See Sections IV and V. ∎

An Efficient Use of Backward Networks: In our model, we assume that the fraction λ of time is used for the purpose of feedback to aid forward-message computation. Here we investigate whether or not using the backward network for feedback can be more beneficial than other possible uses of the backward network. It turns out that for a wide range of channel parameters, the backward network can be more efficiently used rather than if it were used for other purposes. To show this clearly, let us consider an example where $(m,n)=(0,3)$ and $(\tilde{m},\tilde{n})=(2,1)$. Specializing Theorem 1 to this case, we get the capacity

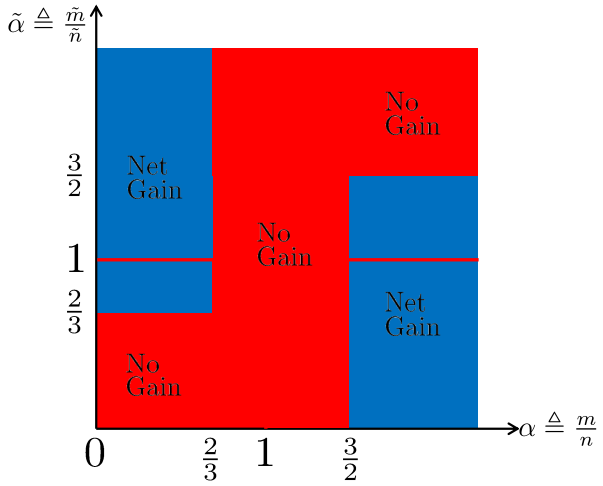

Fig. 3. Net feedback gain in computation.

gain due to the use of the backward network for feedback:

$$\Delta C = \min\left\{\lambda\tilde{m}, \frac{2}{3}n - m\right\} = 2\lambda. \tag{1}$$

We now consider the opportunity cost: the capacity gain due to the use of the backward network for other purposes. One natural alternative in this context is to use the backward network for its own message computation. In this case, the capacity gain is:

$$\Delta C = \lambda C_{\mathsf{no}} = \lambda. \tag{2}$$

From (1) and (2), one can see that the backward network offers larger capacity gain when it is used for the purpose of feedback. Fig. 2. plots the corresponding two capacity gains as a function of λ. Notice that when $\lambda = 1$, the capacity gain due to the use for feedback is 2 bits, while the capacity gain due to the use for backward-message computation is 1 bit.

Furthermore, using the above, one can quantify net feedback gain as follows. We can view the feedback cost as the opportunity cost since it is the capacity gain due to other alternative, meaning the price that should be paid for using the backward network for feedback. Hence the net feedback gain can be quantified as:

Capacity gain due to feedback - opportunity cost.

In the above example, the net feedback gain is $2\lambda - \lambda = \lambda \geq 0$. From the above, one can also see that the strictly positive net gain implies that the backward network is more efficiently used for the purpose of feedback than other purposes. Fig 3. shows the entire channel parameter regimes in which there is net feedback gain.

IV. Proof of Achievability

By symmetry, we consider only the regime of $\alpha \leq 1$. A simple uncoded transmission can yield $R = n$ for the case of $\alpha = 1$. For $\frac{2}{3} \leq \alpha < 1$, the nonfeedback scheme in [13] gives $R = \frac{2}{3}n$. Hence, our focus is the case of $0 \leq \alpha \leq \frac{2}{3}$.

Our achievability proof consists of two parts. We first employ the network decomposition developed in [13] to decompose a forward network into elementary orthogonal subnetworks. We then apply achievable schemes separately for the elementary subnetworks. We will show that these two parts lead us to obtain the desired achievable rate of an original network, as claimed.

A. Achievability via Network Decomposition

Let us first review the network decomposition in [13], for the regime $0 \leq \alpha \leq \frac{2}{3}$ of our interest.

Theorem 2 (Network Decomposition): The (m, n) network can be separated into a combination of the subnetworks as follows.

$$(m,n) \longrightarrow \begin{cases} (0,1)^{n-2m} \times (1,2)^m, & 0 \leq \alpha \leq \frac{1}{2}, \\ (1,2)^{2n-3m} \times (2,3)^{2m-n}, & \frac{1}{2} \leq \alpha \leq \frac{2}{3}. \end{cases}$$

Here we use the symbol $\times$ for the concatenation of orthogonal models, just like in $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$. Theorem 2 implies that fundamental building blocks constitute only three models: $(0, 1)$, $(1, 2)$, and $(2, 3)$. Hence, we focus on these models.

Lemma 1 (Achievable Rates for Elementary Subnetworks): Let $R^{(m,n)}$ be the computation rate of (m, n) model.

$$\begin{aligned} R^{(0,1)} &= \min\left\{\lambda\tilde{m}, \frac{2}{3}\right\}, \\ R^{(1,2)} &= \min\left\{1 + \lambda\tilde{m}, \frac{4}{3}\right\}, \\ R^{(2,3)} &= 2. \end{aligned}$$

Proof: See Section IV-B. ∎

As mentioned above, we focus on the regime of $0 \leq \alpha \leq \frac{2}{3}$. To apply the network decomposition theorem, let us consider two cases: *a)* $0 \leq \alpha < \frac{1}{2}$; *b)* $\frac{1}{2} \leq \alpha \leq \frac{2}{3}$.

a) $0 \leq \alpha \leq \frac{1}{2}$*:* Using the decomposition in Theorem 2, we get:

$$\begin{aligned} R &= (n - 2m) \cdot R^{(0,1)} + m \cdot R^{(1,2)} \\ &= \min\left\{m + \lambda\tilde{m}, \frac{2}{3}n\right\}. \end{aligned} \tag{3}$$

where the second inequality is due to detailed yet straightforward calculation given in Appendix A.

b) $\frac{1}{2} \leq \alpha \leq \frac{2}{3}$*:* From Theorem 2, (m, n) forward network can be decomposed as: $(m, n) \longrightarrow (1, 2)^{2n-3m} \times (2, 3)^{2m-n}$. We use $(\tilde{m}, \tilde{n})$ backward network for the transmission associated with the number $2n - 3m$ of $(1, 2)$ forward networks. We also split the fraction λ of time (assigned to the backward network) equally into each of $2n - 3m$ parts. We then use the split fraction $\frac{\lambda}{2n-3m}$ of the backward network for aiding the transmission w.r.t. each $(1, 2)$ forward network. This gives

$$R^{(1,2)} = \min\left\{1 + \frac{\lambda}{2n - 3m} \cdot \tilde{m}, \frac{4}{3}\right\}.$$

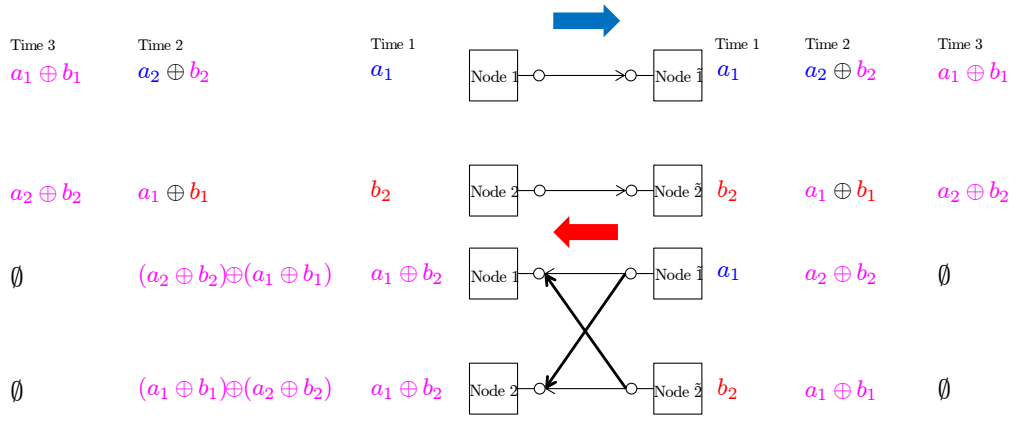

Fig. 4. An achievable scheme for $(m,n)=(0,1)$, $(\tilde{m},\tilde{n})=(1,1)$ and $\lambda=\frac{2}{3}$.

So we get:

$$\begin{aligned} R =& (2n-3m)\cdot R^{(1,2)} + (2m-n)\cdot R^{(2,3)} \\ =& \min\left\{2n-3m+\lambda\tilde{m}, \frac{8n}{3}-4m\right\} + 4m - 2n \\ =& \min\left\{m+\lambda\tilde{m}, \frac{2}{3}n\right\}. \end{aligned}$$

B. Proof of Lemma 1

$(m,n)=(\mathbf{0},\mathbf{1})$, $\forall(\tilde{m},\tilde{n})$: Let us start by reviewing the perfect feedback scheme for the case of $(m,n)=(0,1)$ [9]. While one can readily see that there is no way to compute functions for the nonfeedback case, i.e., $R=0$, feedback can provide a positive rate. The idea is to exploit the following paths with the help of feedback:

[Node 1 $\rightarrow$ Node $\tilde{1}$ $\rightarrow$ feedback $\rightarrow$ Node 2 $\rightarrow$ Node $\tilde{2}$],

[Node 2 $\rightarrow$ Node $\tilde{2}$ $\rightarrow$ feedback $\rightarrow$ Node 1 $\rightarrow$ Node $\tilde{1}$].

The perfect feedback scheme consists of three time slots. In time 1, node 1 and 2 transmit a_1 and b_2 respectively. Node $\tilde{1}$ can then deliver the received symbol a_1 to node 2 through feedback. This symbol can now be used to pre-compute $a_1 \oplus b_1$ at node 2. Similarly, node 1 can pre-compute $a_2 \oplus b_2$. In time 2, node 1 and 2 can forward these pre-computed functions to node $\tilde{1}$ and $\tilde{2}$ respectively. Until the end of time 2, $a_1 \oplus b_1$ is not delivered to node $\tilde{1}$. Similarly $a_2 \oplus b_2$ is missing at node $\tilde{2}$. Using one more time slot, we can deliver these symbols to intended nodes. With feedback, node 1 can get $a_1 \oplus b_1$. Forwarding this at time 3, node $\tilde{1}$ can obtain the $a_1 \oplus b_1$. Similarly node $\tilde{2}$ can get $a_2 \oplus b_2$. Hence node $\tilde{1}$ and $\tilde{2}$ can obtain $(a_1 \oplus b_1, a_2 \oplus b_2)$ during three time slots. This gives a rate of $\frac{2}{3}$.

Our model, however, provides feedback in the *limited* fashion since feedback signals are delivered *only through the backward network*. We next develop a new achievable scheme that uses the backward network efficiently.

We focus on an example network that turns out to play a key role in generalizing into arbitrary values of $(\tilde{m},\tilde{n})$ and λ: $(m,n)=(0,1)$, $(\tilde{m},\tilde{n})=(1,1)$ and $\lambda=\frac{2}{3}$. See Fig. 4. Our achievable scheme consists of three time slots. In time 1, node 1 and 2 deliver a_1 and b_2 respectively. Node $\tilde{1}$ and $\tilde{2}$ then get a_1 and b_2 respectively. Through the backward network, node $\tilde{1}$ and $\tilde{2}$ feed back a_1 and b_2 respectively. Node 1 and 2 then get the same symbol: $a_1 \oplus b_2$. Unlike the perfect feedback scheme, it seems impossible for each node to pre-compute the desired modulo-2 sum from $a_1 \oplus b_2$. However, we can actually pre-compute the desired functions as in the perfect feedback scheme. Specifically node 1 and 2 can pre-compute $a_2 \oplus b_2$ and $a_1 \oplus b_1$ respectively. The key idea is to exploit the previously transmitted symbols as side information. Exploiting a_1 as side information, node 1 can decode b_2 from $a_1 \oplus b_2$, thus obtaining $a_2 \oplus b_2$. Similarly, node 2 can obtain $a_1 \oplus b_1$. Forwarding these symbols at time 2, node $\tilde{1}$ and $\tilde{2}$ can get $a_2 \oplus b_2$ and $a_1 \oplus b_1$ respectively.

As before, node $\tilde{1}$ and $\tilde{2}$ simply feed back the received symbols $a_2 \oplus b_2$ and $a_1 \oplus b_1$ respectively. Then node 1 and 2 get $(a_1 \oplus b_1) \oplus (a_2 \oplus b_2)$. Node 1 can now exploit the transmitted symbol at time 2 $(a_2 \oplus b_2)$ to get $a_1 \oplus b_1$. Similarly, node 2 can get $a_2 \oplus b_2$ using $a_1 \oplus b_1$. Forwarding these two symbols at time 3, node $\tilde{1}$ and $\tilde{2}$ get $a_1 \oplus b_1$ and $a_2 \oplus b_2$ respectively. In summary, node $\tilde{1}$ and $\tilde{2}$ can obtain $(a_1 \oplus b_1, a_2 \oplus b_2)$ during three time slots. This gives $R^{(0,1)} = \frac{2}{3}$.

A generalization into arbitrary values of $(\tilde{m},\tilde{n})$ and λ is described in Appendix II. This yields the desired result:

$$R^{(0,1)} = \min\left\{\lambda\tilde{m}, \frac{2}{3}\right\}. \tag{4}$$

$(m,n)=(\mathbf{1},\mathbf{2})$, $\forall(\tilde{m},\tilde{n})$: We focus on an example network that turns out to play a key role in generalization: $(m,n)=(1,2)$, $(\tilde{m},\tilde{n})=(1,1)$ and $\lambda=\frac{1}{3}$. See Fig. 5. Our scheme consists of three time slots. In time 1, node 1 sends a_1 and a_2; node 2 sends b_2 and b_1. We then achieve $a_2 \oplus b_2$ at node $\tilde{1}$. Similarly we achieve $a_1 \oplus b_1$ at node $\tilde{2}$. Observe that the bottom level at node $\tilde{1}$ and $\tilde{2}$ naturally form the modulo-2 sum function of interest. In time 2, we repeat this w.r.t. new symbols, thus achieving $a_4 \oplus b_4$ and $a_3 \oplus b_3$ at node $\tilde{1}$ and $\tilde{2}$ respectively. Note that until the end of time 2, $(a_1 \oplus b_1, a_3 \oplus b_3)$ are not delivered yet to node

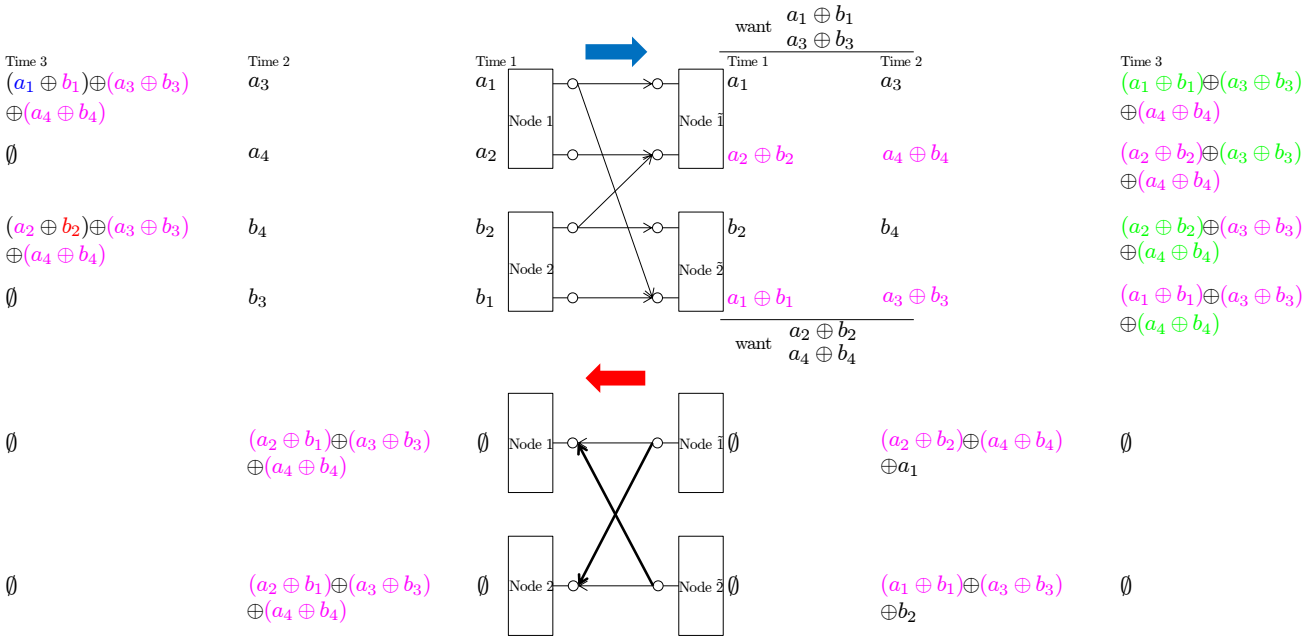

Fig. 5. An achievable scheme for $(m, n) = (1, 2)$, $(\tilde{m}, \tilde{n}) = (1, 1)$ and $\lambda = \frac{1}{3}$.

$\tilde{1}$. Similarly $(a_2 \oplus b_2, a_4 \oplus b_4)$ are missing at node $\tilde{2}$. With feedback, however, we can accomplish the transmission of these signals very efficiently.

Through the backward network, node $\tilde{1}$ and $\tilde{2}$ simultaneously feed back the following symbols at the end of time 2:

$$\text{node } \tilde{1}\text{: } (a_2 \oplus b_2) \oplus (a_4 \oplus b_4) \oplus a_1,$$
$$\text{node } \tilde{2}\text{: } (a_1 \oplus b_1) \oplus (a_3 \oplus b_3) \oplus b_2.$$

Node 1 and 2 then get: $(a_2 \oplus b_1) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4)$. One way to achieve modulo-2 sums at node $\tilde{1}$ and $\tilde{2}$ is to pre-compute the functions and forward these functions. However, the exact pre-computation of the desired modulo-2 sums from $(a_2 \oplus b_1) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4)$ seems impossible. Hence, instead of striving to extract the desired modulo-2 sums, we take a new approach as follows. Exploiting (a_1, a_2) and (b_1, b_2) at node 1 and 2 respectively, node 1 and 2 can compute the following:

$$\text{node 1: } (a_1 \oplus b_1) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4),$$
$$\text{node 2: } (a_2 \oplus b_2) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4).$$

In time 3, node 1 and 2 forward these two symbols only on the top level. Node $\tilde{1}$ and $\tilde{2}$ then get the two symbols. Notice that these two symbols contain the desired functions. Now the key to observe is that node $\tilde{1}$ and $\tilde{2}$ already had $(a_2 \oplus b_2, a_4 \oplus b_4)$ and $(a_1 \oplus b_1, a_3 \oplus b_3)$ respectively. These past received symbols can now be exploited to help obtaining the desired $a_i \oplus b_i$'s that node $\tilde{1}$ and $\tilde{2}$ could not compute before. Specifically, node $\tilde{1}$ can decode $a_3 \oplus b_3$ from $(a_2 \oplus b_2) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4)$ using $(a_2 \oplus b_2, a_4 \oplus b_4)$. Subsequently, exploiting $(a_3 \oplus b_3, a_4 \oplus b_4)$, node $\tilde{1}$ can also decode $a_1 \oplus b_1$ from the received symbol $(a_1 \oplus b_1) \oplus (a_3 \oplus b_3) \oplus (a_4 \oplus b_4)$. Similarly, node $\tilde{2}$ can obtain $a_2 \oplus b_2$ and $a_4 \oplus b_4$. Here the key observation is that node $\tilde{1}$ and $\tilde{2}$ exploit past received symbols as side information to aid computation. As a result, node $\tilde{1}$ and $\tilde{2}$ can obtain $a_i \oplus b_i$'s for $i = 1, \cdots, 4$, during three time slots, thus achieving $R^{(1,2)} = \frac{4}{3}$.

We find that this idea can be extended to arbitrary values of $(\tilde{m}, \tilde{n})$ and λ. With this, we can get the desired result:

$$R^{(1,2)} = \min\left\{1 + \lambda\tilde{m}, \frac{4}{3}\right\}. \tag{5}$$

$(m, n) = (\mathbf{2}, \mathbf{3})$, $\forall(\tilde{m}, \tilde{n})$: Using the nonfeedback scheme [13], we get $R^{(2,3)} = 2$.

V. Proof of Converse

The proof for the case of $\alpha = 1$ is straightforward due to the standard cut-set argument: $N(R - \epsilon_N) \leq I(S_1^K \oplus S_2^K; Y_1^N) \leq \sum H(Y_{1i}) \leq N \max(m, n)$. If R is achievable, then $\epsilon_N \to 0$ as N tends to infinity, and hence $R \leq \max(m, n) = n$. For the case of $\alpha \neq 1$, one can see that it suffices to prove the following bounds:

$$R \leq \min\left\{m + \lambda\tilde{m}, n + \lambda\tilde{n}, \frac{2}{3}\max(m, n)\right\}.$$

Note that the third bound matches the perfect feedback bound [9]. Hence it is also an outer bound of our network. We include the proof of the first bound of $m + \lambda\tilde{m}$ as below. By symmetry, the proof of the second bound can be derived in a similar manner.

Starting with Fano's inequality, we get

$$\begin{aligned}
N(R-\epsilon_N) &\le I(S_1^K \oplus S_2^K; Y_1^N) \\
&\overset{(a)}{\le} I(S_1^K \oplus S_2^K; Y_1^N, S_1^K) \\
&\overset{(b)}{=} H(Y_1^N|S_1^K) \\
&\le H(Y_1^N, \tilde{V}_2^N|S_1^K) \\
&= \sum H(Y_{1i}, \tilde{V}_{2i}|S_1^K, Y_1^{i-1}, \tilde{V}_2^{i-1}) \\
&\overset{(c)}{=} \sum H(Y_{1i}, \tilde{V}_{2i}|S_1^K, Y_1^{i-1}, \tilde{V}_2^{i-1}, \tilde{X}_1^i, \tilde{Y}_1^{i-1}, X_{1i}) \\
&\le \sum [H(Y_{1i}|X_{1i}) + H(\tilde{V}_{2i})] \\
&= \sum [H(V_{2i}) + H(\tilde{V}_{2i})] \\
&\overset{(d)}{\le} N(m + \lambda\tilde{m})
\end{aligned}$$

where (a) follows from the nonnegativity of mutual information; (b) follows from the independence of S_1^K and S_2^K; (c) follows from the fact that $\tilde{X}_1^i$ is a function of Y_1^{i-1} and X_{1i} is a function of $(S_1^K, \tilde{Y}_1^{i-1})$; (d) follows from $\sum H(\tilde{V}_{2i}) \le N\lambda\tilde{m}$. If R is achievable, then $\epsilon_N \to 0$ as N tends to infinity. Hence we get $R \le m + \lambda\tilde{m}$.

VI. Conclusion

For the four-node ADT deterministic network, we developed a new achievable scheme and derived upper bounds, thereby establishing feedback computation capacity. Our achievable scheme takes a separation approach based on a network decomposition framework. Our future work is along several new directions: (1) Generalizing to four-source scenarios in which two nodes that transmitted the forward messages also wish to compute a function of two additional backward messages generated from the other two nodes; (2) extending to arbitrary multi-hop networks [15], [16].

Acknowledgement

The authors thank Prof. Michael Gastpar for discussions on the optimality of network decomposition in [13]. This work was supported by ICT R&D program of MSIP/IITP. [1391104004, Development of Device Collaborative Giga-Level Smart Cloudlet Technology]

Appendix I
Derivation of (3)

By Theorem 2, for the case where $0 \le \alpha \le \frac{1}{2}$, (m, n) forward network can be decomposed as:

$$(m, n) \longrightarrow (0,1)^{n-2m} \times (1,2)^m.$$

Let

$$k = \frac{n-2m}{n-m} + \frac{\frac{1}{3}nm - \frac{2}{3}m^2}{(\frac{2}{3}n - m)(n-m)}. \tag{6}$$

Note that $0 \le k \le 1$. It will be clearer as to why k is set as above. Among all the values of λ, during $k\lambda$, we use $(\tilde{m}, \tilde{n})$ backward network for the transmission associated with the number $n-2m$ of $(0,1)$ forward networks. The next step is that we split the fraction $k\lambda$ of time equally into each of $n-2m$ parts. We then use the split fraction $\frac{k\lambda}{n-2m}$ of the backward network to help the transmission w.r.t. each $(0,1)$ forward network. This gives

$$R^{(0,1)} = \min\left\{\frac{k\lambda}{n-2m} \cdot \tilde{m}, \frac{2}{3}\right\}.$$

Similarly, for the remaining $(1-k)\lambda$, we use the backward network for the transmission associated with $(1,2)^m$ forward networks. We also split the fraction $(1-k)\lambda$ of time equally into each of m parts. So we use the split fraction $\frac{(1-k)\lambda}{m}$ of the backward network to aid the transmission w.r.t. each $(1,2)$ forward network. This gives

$$R^{(1,2)} = \min\left\{1 + \frac{(1-k)\lambda}{m} \cdot \tilde{m}, \frac{4}{3}\right\}.$$

Hence we get:

$$\begin{aligned}
R =& (n-2m) \cdot R^{(0,1)} + m \cdot R^{(1,2)} \\
=& \min\Bigg\{m + \lambda\tilde{m}, k\lambda\tilde{m} + \frac{4}{3}m, \\
& \quad m + (1-k)\lambda\tilde{m} + \frac{2}{3}(n-2m), \frac{2}{3}n\Bigg\}.
\end{aligned} \tag{7}$$

Now it suffices to show that (7) becomes $\min\left\{m + \lambda\tilde{m}, \frac{2}{3}n\right\}$. To show this, let us divide into two cases.

A. $m + \lambda\tilde{m} \ge \frac{2}{3}n$ (*i.e.,* $\lambda\tilde{m} \ge \frac{2}{3}n - m$)

In this case, one can readily get:

$$\begin{aligned}
& k\lambda\tilde{m} + \frac{4}{3}m - \frac{2}{3}n \\
& \ge k\left(\frac{2}{3}n - m\right) + \frac{4}{3}m - \frac{2}{3}n \\
& \overset{(a)}{=} \frac{1}{n-m}\left\{(n-2m)\left(\frac{2}{3}n - m\right) + \frac{1}{3}nm - \frac{2}{3}m^2\right\} \\
& + \frac{4}{3}m - \frac{2}{3}n \\
& = \frac{1}{n-m}\left\{\frac{2}{3}n^2 - 2nm + \frac{4}{3}m^2\right\} + \frac{4}{3}m - \frac{2}{3}n \\
& = \frac{1}{n-m}\left\{\frac{2}{3}n^2 - \frac{2}{3}nm\right\} - \frac{2}{3}n \\
& = 0,
\end{aligned}$$

where (a) follows from (6). Using (6), we also get:

$$\begin{aligned}
& m + (1-k)\,\lambda\tilde{m} + \frac{2}{3}(n-2m) - \frac{2}{3}n \\
& \ge (1-k)\left(\frac{2}{3}n - m\right) - \frac{1}{3}m \\
& = \frac{1}{n-m}\left\{m\left(\frac{2}{3}n - m\right) - \frac{1}{3}nm + \frac{2}{3}m^2\right\} - \frac{1}{3}m \\
& = \frac{1}{n-m}\left\{\frac{1}{3}nm - \frac{1}{3}m^2\right\} - \frac{1}{3}m \\
& = 0.
\end{aligned}$$

From the two inequalities as above, one can see $R = \frac{2}{3}n$.

B. $m + \lambda\tilde{m} < \frac{2}{3}n$ *(i.e.,* $\lambda\tilde{m} < \frac{2}{3}n - m$*)*

In this case, one can easily get:

$$\begin{aligned}
&k\lambda\tilde{m} + \frac{4}{3}m - (m + \lambda\tilde{m}) \\
&= \frac{1}{3}m + (k-1)\lambda\tilde{m} \\
&\overset{(a)}{>} \frac{1}{3}m + (k-1)\left(\frac{2}{3}n - m\right) \\
&\overset{(b)}{=} \frac{1}{3}m + \frac{1}{n-m}\left\{\frac{1}{3}nm - \frac{2}{3}m^2 - m\left(\frac{2}{3}n - m\right)\right\} \\
&= 0,
\end{aligned}$$

where (a) follows from the fact that $0 \leq k \leq 1$; and (b) follows from (6). Similarly, we can also get:

$$\begin{aligned}
&m + (1-k)\lambda\tilde{m} + \frac{2}{3}(n-2m) - (m + \lambda\tilde{m}) \\
&= -k\lambda\tilde{m} + \frac{2}{3}(n-2m) \\
&> -k\left(\frac{2}{3}n - m\right) + \frac{2}{3}(n-2m) \\
&= \frac{1}{n-m}\left\{-(n-2m)\left(\frac{2}{3}n - m\right) - \frac{1}{3}nm + \frac{2}{3}m^2\right\} \\
&+ \frac{2}{3}(n-2m) \\
&= \frac{1}{n-m}\left\{-\frac{2}{3}n^2 + 2nm - \frac{4}{3}m^2\right\} + \frac{2}{3}(n-2m) \\
&= 0,
\end{aligned}$$

From the two inequalities as above, one can see $R = m + \lambda\tilde{m}$.

Consequently, from the cases *A* and *B*, we can conclude that (7) becomes

$$\min\left\{m + \lambda\tilde{m}, \frac{2}{3}n\right\}.$$

This completes the proof of (3).

Appendix II
Proof of (4)

We provide a complete proof of (4) for arbitrary values of $(\tilde{m}, \tilde{n})$ and λ. We focus on the case of $\tilde{\alpha} \leq 1$. The other case similarly follows.

A. Achievability

Lemma 2: Let $(m, n) = (0, 1)$ and $(\tilde{m}, \tilde{n}) = (\tilde{r}, \tilde{r}+1)$. In time i, suppose node 1 and 2 transmit a_{2i-1} and b_{2i} respectively, $i = 1, \cdots, \tilde{r}$. During the next $2\tilde{r}$ time slots, node $\tilde{1}$ and $\tilde{2}$ can obtain $a_j \oplus b_j$, $j = 1, \cdots, 2\tilde{r}$. This can be achieved through using the backward network twice.

Proof: See Appendix II-B. ∎

Lemma 3: Let $(m, n) = (0, 1)$, $\forall(\tilde{m}, \tilde{n})$. In time i, suppose node 1 and 2 send a_{2i-1} and b_{2i} respectively, $i = 1, \cdots, \tilde{m}$. During the next $2\tilde{m}$ time slots, node $\tilde{1}$ and $\tilde{2}$ can obtain $a_j \oplus b_j$, $j = 1, \cdots, 2\tilde{m}$. This can be achieved through using the backward network twice.

Proof: We first apply *Network decomposition theorem* in [13] for $(\tilde{m}, \tilde{n})$ backward network. Then the network can be separated as follows.

$$(\tilde{m}, \tilde{n}) \longrightarrow (\tilde{r}, \tilde{r}+1)^{\tilde{n}-\tilde{m}-\tilde{a}} \times (\tilde{r}+1, \tilde{r}+2)^{\tilde{a}},$$

where

$$\begin{aligned}
\tilde{r} &= \left\lfloor \frac{\tilde{m}}{\tilde{n}-\tilde{m}} \right\rfloor, \\
\tilde{a} &= \tilde{m} \mod \tilde{n} - \tilde{m}.
\end{aligned}$$

We now apply Lemma 2 for the number $\tilde{n}-\tilde{m}-\tilde{a}$ of $(\tilde{r}, \tilde{r}+1)$ subnetworks and the number $\tilde{a}$ of $(\tilde{r}+1, \tilde{r}+2)$ subnetworks. Since

$$(\tilde{n} - \tilde{m} - \tilde{a}) \cdot 2\tilde{r} + \tilde{a} \cdot 2(\tilde{r}+1) = 2(\tilde{r}(\tilde{n}-\tilde{m}) + a) = 2\tilde{m},$$

we can verify that node $\tilde{1}$ and $\tilde{2}$ can obtain $a_j \oplus b_j$, $j = 1, \cdots, 2\tilde{m}$ using each subnetwork twice (hence using $(\tilde{m}, \tilde{n})$ backward network twice). Note that we use additional $2\tilde{m}$ time slots: $(\tilde{n} - \tilde{m} - \tilde{a}) \cdot 2\tilde{r} + \tilde{a} \cdot 2(\tilde{r}+1) = 2(\tilde{r}(\tilde{n}-\tilde{m}) + a) = 2\tilde{m}$. ∎

With Lemma 3, we now show $R^{(0,1)} = \min\left\{\lambda\tilde{m}, \frac{2}{3}\right\}$. To do this, let us divide into two subcases.

1) $\lambda\tilde{m} > \frac{2}{3}$*:* In this case, for the first $\tilde{m}$ time slots, node 1 and 2 deliver a_{2i-1} and b_{2i} respectively, $i = 1, \cdots, \tilde{m}$. By Lemma 3, for the next $2\tilde{m}$ time slots, node $\tilde{1}$ and $\tilde{2}$ can obtain $a_j \oplus b_j$, $j = 1, \cdots, 2\tilde{m}$. As the total time spent in the forward network is $3\tilde{m}$ and $\lambda \cdot 3\tilde{m} > 2$, it is guaranteed to use the backward network twice. The computation rate for $(0, 1)$ network is therefore

$$R^{(0,1)} = \frac{0 + 2\tilde{m}}{\tilde{m} + 2\tilde{m}} = \frac{2}{3}.$$

2) $\lambda\tilde{m} \leq \frac{2}{3}$*:* Our strategy in this case is: For the first $\tilde{m}$ time slots, node 1 and 2 deliver a_{2i-1} and b_{2i} respectively, $i = 1, \cdots, \tilde{m}$. By Lemma 3, for the next $2\tilde{m}$ time slots, node $\tilde{1}$ and $\tilde{2}$ can obtain $a_j \oplus b_j$, $j = 1, \cdots, 2\tilde{m}$. Notice that this can be achieved through using the backward network twice. However, as the total time spent in the forward network is $3\tilde{m}$ and $\lambda \cdot 3\tilde{m} \leq 2$, it is not guaranteed to use the backward network twice yet. We solve this by waiting for the next l time slots. Here we set $l := \frac{2}{\lambda} - 3\tilde{m}$. Then we now guarantee to use the backward network twice because $\lambda \cdot (3\tilde{m} + l) = 2$. This yields the computation rate of

$$R^{(0,1)} = \frac{0 + 2\tilde{m} + 0}{\tilde{m} + 2\tilde{m} + l} = \lambda\tilde{m}.$$

From these two subcases, we verify that $R^{(0,1)} = \min\left\{\lambda\tilde{m}, \frac{2}{3}\right\}$.

B. Proof of Lemma 2

If node 1 and 2 send a_{2i-1} and b_{2i} respectively at time i, $i = 1, \cdots, \tilde{r}$, node $\tilde{1}$ and $\tilde{2}$ get a_{2i-1} and b_{2i} respectively. One can readily see that until the end of time $\tilde{r}$, there is no way to compute the modulo-2 sum functions. We will now explain how node $\tilde{1}$ and $\tilde{2}$ can obtain $a_k \oplus b_k$, $k = 1, \cdots, 2\tilde{r}$ for the next $2\tilde{r}$ time slots.

Node $\tilde{1}$ feeds back symbols, a_1 to $a_{2\tilde{r}-1}$, through the signal bit levels and each symbol is sent to each level respectively from the top. Similarly, node $\tilde{2}$ feeds back b_{2i} for $i = 1, 2, \cdots, \tilde{r}$. However, there are $\tilde{r}+1$ signal bit levels in total. Hence, we note that node $\tilde{1}$ and $\tilde{2}$ send nothing through the lowest signal bit level ($(\tilde{r}+1)$th level).

Let us first consider symbols that come through the bottom level at node 1. Since node $\tilde{1}$ and $\tilde{2}$ feed back nothing on the bottom level of $(\tilde{r}, \tilde{r}+1)$ backward network, node 1 gets $b_{2\tilde{r}}$ that comes *only* through the cross link between node 1 and node $\tilde{2}$. Similarly, node 2 gets $a_{2\tilde{r}-1}$ on the bottom level. Exploiting $a_{2\tilde{r}}$ as side information, node 1 can now get $a_{2\tilde{r}} \oplus b_{2\tilde{r}}$. Similarly, node 2 can get $a_{2\tilde{r}-1} \oplus b_{2\tilde{r}-1}$. Forwarding these symbols at time $\tilde{r}+1$, node $\tilde{1}$ and $\tilde{2}$ obtain $a_{2\tilde{r}} \oplus b_{2\tilde{r}}$ and $a_{2\tilde{r}-1} \oplus b_{2\tilde{r}-1}$ respectively.

Let us now consider symbols that come through the $\tilde{r}-1$ signal bit levels among the remaining $\tilde{r}$ signal bit levels. This excludes the symbols from the top level. From the 2nd level to $\tilde{r}$th level, node 1 and 2 get:

$$\text{node 1: } a_{2j-1} \oplus b_{2(j-1)},$$
$$\text{node 2: } a_{2(j-1)-1} \oplus b_{2j},$$

$j = 2, \cdots, \tilde{r}$. Node 1 and 2 then exploit $(a_{2j-1}, a_{2(j-1)})$ and $(b_{2(j-1)-1}, b_{2j})$ to get $a_{2(j-1)} \oplus b_{2(j-1)}$ and $a_{2(j-1)-1} \oplus b_{2(j-1)-1}$ respectively. In time $\tilde{r}+j$, node 1 and 2 transmit these two symbols through $(0,1)$ forward network. Node $\tilde{1}$ and $\tilde{2}$ then obtain $a_{2(j-1)} \oplus b_{2(j-1)}$ and $a_{2(j-1)-1} \oplus b_{2(j-1)-1}$ respectively, $j = 2, \cdots, \tilde{r}$.

The next step is that node $\tilde{1}$ and $\tilde{2}$ simply feed back what they received: Node $\tilde{1}$ sends symbols, $a_2 \oplus b_2$ to $a_{2\tilde{r}} \oplus b_{2\tilde{r}}$, through the signal bit levels and each symbol is sent to each level respectively from the top; similarly, node $\tilde{2}$ feeds back $a_{2i-1} \oplus b_{2i-1}$ for $i = 1, 2, \cdots, \tilde{r}$.

Consider symbols that come through the bottom level at node 1 and 2. Since node $\tilde{1}$ and $\tilde{2}$ feed back nothing on the bottom level, node 1 and 2 get $a_{2\tilde{r}-1} \oplus b_{2\tilde{r}-1}$ and $a_{2\tilde{r}} \oplus b_{2\tilde{r}}$ respectively. Forwarding these symbols at time $2\tilde{r}+1$, node $\tilde{1}$ and $\tilde{2}$ get $a_{2\tilde{r}-1} \oplus b_{2\tilde{r}-1}$ and $a_{2\tilde{r}} \oplus b_{2\tilde{r}}$ respectively.

Now consider symbols that come through the $\tilde{r}-1$ signal bit levels among the remaining $\tilde{r}$ signal bit levels. This excludes the symbols from the top level. From the 2nd level to $\tilde{r}$th level, node 1 and 2 get:

$$\text{node 1: } (a_{2j} \oplus b_{2j}) \oplus (a_{2(j-1)-1} \oplus b_{2(j-1)-1}),$$
$$\text{node 2: } (a_{2(j-1)} \oplus b_{2(j-1)}) \oplus (a_{2j-1} \oplus b_{2j-1}),$$

$j = 2, \cdots, \tilde{r}$. Node 1 can now exploit the previously transmitted symbol $a_{2j} \oplus b_{2j}$ to get $a_{2(j-1)-1} \oplus b_{2(j-1)-1}$; similarly node 2 gets $a_{2(j-1)} \oplus b_{2(j-1)}$, $j = 2, \cdots, \tilde{r}$. Forwarding these two symbols at time $2\tilde{r}+j$, node $\tilde{1}$ and $\tilde{2}$ get $a_{2(j-1)-1} \oplus b_{2(j-1)-1}$ and $a_{2(j-1)} \oplus b_{2(j-1)}$ respectively, $j = 2, \cdots, \tilde{r}$.

In summary, node $\tilde{1}$ and $\tilde{2}$ obtain $a_k \oplus b_k$, $k = 1, \cdots, 2\tilde{r}$. Notice that we use the backward network twice. This completes the proof of Lemma 2.

REFERENCES

[1] C. E. Shannon, "The zero error capacity of a noisy channel," *IRE Transactions on Information Theory*, vol. 2, pp. 8–19, Sept. 1956.

[2] T. M. Cover and S. Pombra, "Gaussian feedback capacity," *IEEE Transactions on Information Theory*, vol. 35, pp. 37–43, Jan. 1989.

[3] Y.-H. Kim, "Feedback capacity of the first-order moving average Gaussian channel," *IEEE Transactions on Information Theory*, vol. 52, pp. 3063–3079, July 2006.

[4] N. T. Gaarder and J. K. Wolf, "The capacity region of a multiple-access discrete memoryless channel can increase with feedback," *IEEE Transactions on Information Theory*, Jan. 1975.

[5] L. H. Ozarow, "The capacity of the white Gaussian multiple access channel with feedback," *IEEE Transactions on Information Theory*, vol. 30, no. 4, pp. 623–629, July 1984.

[6] L. H. Ozarow and S. K. Leung-Yan-Cheong, "An achievable region and outer bound for the Gaussian broadcast channel with feedback," *IEEE Transactions on Information Theory*, vol. 30, pp. 667–671, 1984.

[7] C. Suh and D. Tse, "Feedback capacity of the Gaussian interference channel to within 2 bits," *IEEE Transactions on Information Theory*, vol. 57, pp. 2667–2685, May 2011.

[8] C. Suh, I.-H. Wang, and D. Tse, "Two-way interference channels," *Proc. IEEE International Symposium on Information Theory*, July 2012.

[9] C. Suh and M. Gastpar, "Interactive function computation," *Proc. IEEE International Symposium on Information Theory*, July 2013.

[10] A. S. Avestimehr, S. N. Diggavi, and D. Tse, "Wireless network information flow: A deterministic approach," *IEEE Transactions on Information Theory*, vol. 57, pp. 1872–1905, Apr. 2011.

[11] G. Bresler and D. Tse, "The two-user Gaussian interference channel: a deterministic view," *European Transactions on Telecommunications*, vol. 19, pp. 333–354, June 2008.

[12] J. Zhan, S. Y. Park, M. Gastpar, and A. Sahai, "Function compuation in networks: Duality and constant gap results," in *Proc. 49th Annu. Allerton Conf. Communication, Control, and Computing*, Sep. 2011.

[13] C. Suh, N. Goela, and M. Gastpar, "Computation in multicast networks: Function alignment and converse theorems," *Submitted to the IEEE Transactions on Information Theory*.

[14] C. Suh and M. Gastpar, "Network decomposition for function computation," *IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2013.

[15] A. Ramamoorthy and M. Langberg, "Communicating the sum of sources over a network," *arXiv:1001.5319*, Jan. 2010.

[16] B. Rai and B. Dey, "On network coding for sum-networks," *IEEE Transactions on Information Theory*, vol. 58, pp. 50–63, Jan. 2012.