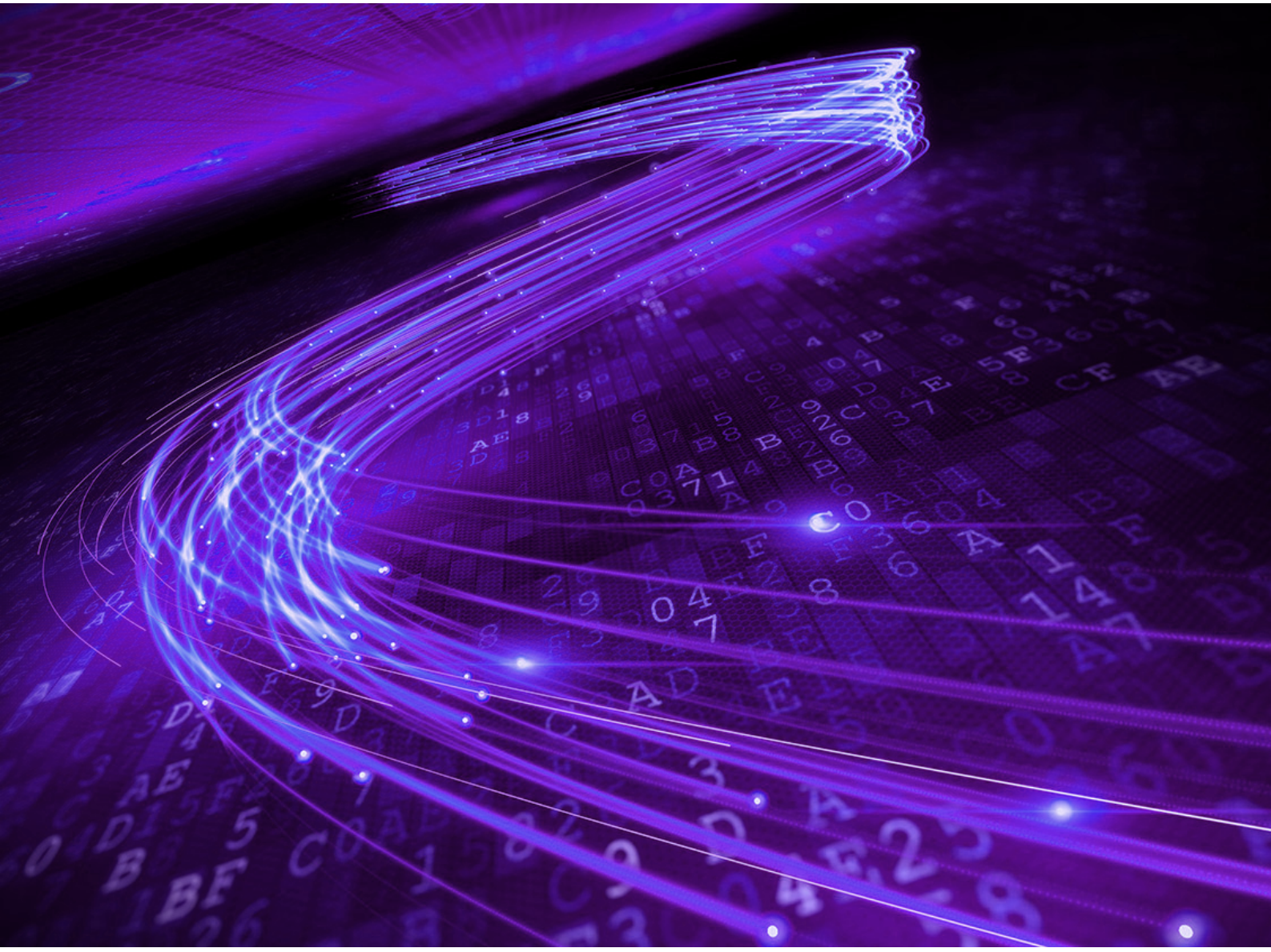
White Paper

# Network Coding, Distributed Data Storage, and DPU

by Raymond W. Yeung



*About the Author*

The author co-founded the field of network coding at the turn of the last century that has induced a paradigm shift in network communications. For his research contributions, he is recognized with the highest honors in information sciences, including the IEEE Richard W. Hamming Medal and the Claude E. Shannon Award. He is a founder of n-hop technologies that pioneers the applications of network coding. He is a Fellow of IEEE and National Academy of Inventors of the USA.


## Abstract

This white paper discusses the potential of a network coding (NC) accelerator inside a data processing unit (DPU). Its main applications would be in a variety of distributed data storage systems found in data centers and content distribution networks (CDN). We start by giving a very high-level introduction to network coding. Using CDN as an example, we point out that the application of network coding can bring very significant benefits to distributed storage systems in terms of storage and bandwidth savings. This justifies an NC accelerator inside a DPU because the application calls for very high-speed data processing (encoding, recoding, and decoding for network coding). A NC-integrated DPU can very well be a game changer for the CDN market in Greater China and beyond.


# Network Coding

Consider transmitting a message consisting of data packets in a computer network. Traditionally, this is achieved by routing the data packets from the source node to the destination node, where at an intermediate node a router receives the data packets from one input link and routes them to one of the output links without modification. At the turn of the last century, the author with his collaborators developed in his seminal works [1]-[4] a new paradigm for network communication by allowing for coding (data processing in the form of mathematical operations) of data packets inside the network. Instead of just routing, the received data packets received at an intermediate node can be combined into new data packets that are transmitted to the next node. This data processing inside the network is called network coding.

In network communication,

- multicast means sending a message from a source node to multiple destination nodes;
- multi-path means sending a message from a source node to the destination node through more than one transmission path;
- multi-hop means sending a message from a source node to the destination node by going through multiple communication links.

The benefit of network coding can be realized in multicast, multi-path, multi-hop, or any combination of these. In multi-hop, network coding can prevent the accumulation of packet loss along the transmission path.

In order to quantify the advantage of network coding over routing, we define

$$Network\ coding\ gain = \frac{Network\ coding\ throughput}{Routing\ throughput}.$$

It was first shown in [1] that the network coding gain can exceed 1. In other words, by employing network coding, more information can be sent through the network. More on network coding gain in next section.

Network coding can bring benefit to any system that can be modeled as a network. Since 2000, it has been developed into a cross-disciplinary field of research. More than 10,000 papers have been written on network coding, including its application to many different domains such as computer communication, wireless network, distributed data storage, etc.

# Distributed Data Storage

Distributed data storage is found in data centers, peer-to-peer storage systems, content delivery networks (CDN), etc. The application of network coding to distributed storage can bring benefit in terms of storage saving, reliability, access delay, and speed of recovery, which translates into reduced CAPEX and OPEX. In this section, we use CDN as an example to illustrate the advantage of network coding for storage saving.

Content Delivery Network (CDN) is a geographically distributed network of proxy servers and their data centers. It alleviates the performance bottleneck of the Internet by providing high availability and performance by distributing the service spatially relative to the end users. CDN is the dominating technology for massive content distribution such as video streaming, social network, software update, etc.

In a traditional CDN, popular contents are stored on edge servers that are geographically close to the end users. To download such contents, the end user only needs to access a local edge server instead of the central server in the cloud. Which edge server the end user chooses to access depends on the distances to the servers, the traffic condition in the network, the accessibility of the servers, etc. In general, a CDN reduces bandwidth costs, improves page load times, and increases the global availability of content.

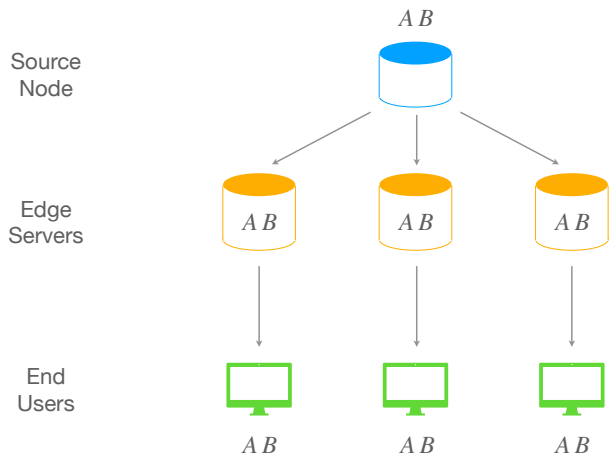



Fig. 1 A traditional CDN with 3 edge servers, where an end user can download the file by accessing any 1 of the 3 edge servers. Two source symbols $A$ and $B$ are delivered to the end user.

Fig. 1 illustrates a traditional CDN. Here, the file to be delivered, which is generated at the source node, consists of 2 symbols $A$ and $B$, called the source symbols. Both $A$ and $B$ are stored on every edge server, and an end user can download the file by accessing any one of the edge servers. Conceptually, the file is multicast to 3 logical end users through the network, where each of them can access a different edge server.

For better performance, a CDN may allow an end user to access more than one edge server. This is illustrated in Fig. 2, in which the file consists of 3 source symbols $A$, $B$, and $C$. Since an end user can access 2 edge servers, it is not necessary to store all $A$, $B$, and $C$ on every edge server. As a result, while each edge server continues to store 2 symbols, 3 source symbols can be delivered to the end user.

Conceptually, the file is multicast to 3 logical end users through the network, where each of them can access a different subset of 2 edge servers.

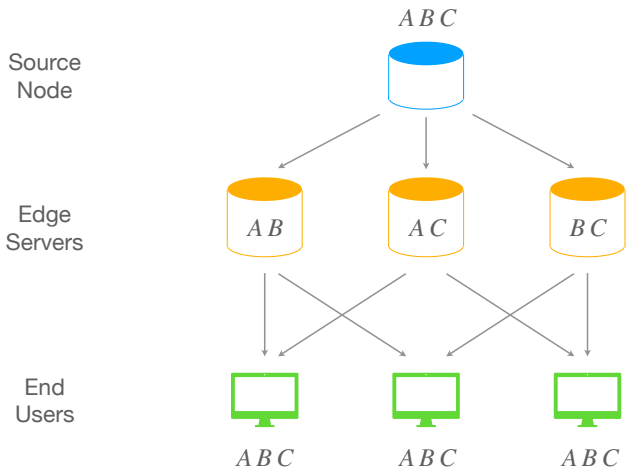



Fig. 2 A CDN with 3 edge servers, where an end user can download the file by accessing any 2 of the 3 edge servers. Three source symbols $A$, $B$, and $C$ are delivered to the end user.

In both Figs. 1 and 2, the source symbols are stored on the edge servers without modification. As such, the solutions therein can be understood as routing solutions, namely that the source symbols are routed to the end users through the edge servers.

The performance of the CDN can be further improved by network coding. In Fig. 3, the file to be delivered consists of 4 symbols $A$, $B$, $C$, and $D$. At the right edge node, coded versions of the source symbols, namely $A+B$ and $C+D$, are stored, viz. network coding. By accessing the left and the right edge servers, the user can decode $C$ from $A$ and $A+C$, and decode $D$ from $B$ and $B+D$. This way, while each edge server continues to store 2 symbols, 4 symbols can be delivered to the end user. In general, network coding can achieve the maximum possible throughput for any given network.

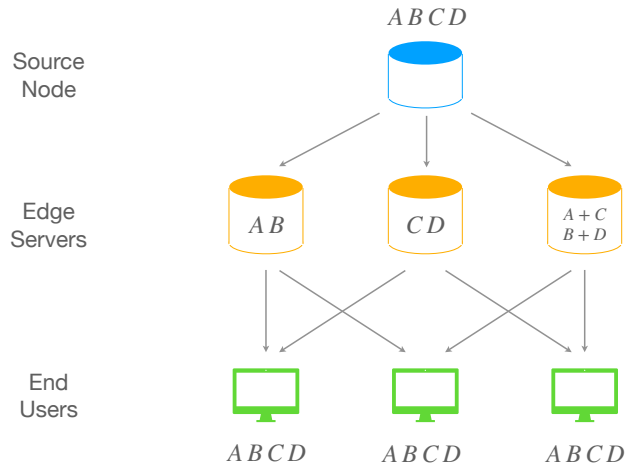



Fig. 3 A CDN with 3 edge servers, where an end user can download the file by accessing any 2 of the 3 edge servers. Network coding is applied. Four source symbols $A$, $B$, $C$, and $D$ are delivered to the end user.

The network in Figs. 2 and 3 is called the (3,2) combination network, in which there are 3 edge servers and the logical end users can access distinct subsets of 2 edge servers. For the (3,2) combination network, the network throughput is 4 while the routing throughput is 3. Therefore, the network coding gain is 4/3 = 1.33.

The network coding gain can be translated into storage saving as follows. If we want to deliver 4 symbols to the end users using the routing solution in Fig. 2, we need to expand the storage at each edge server by 1/3. In other words, the storage saving is 25%. Concomitantly, the capacity of the link connecting the source node to each edge server also needs to be expanded by 1/3. In general,

$$Storage\ saving = \left(1 - \frac{1}{Network\ coding\ gain}\right) \times\ 100\%.$$

In [5], the author and his student studied the network coding gain of a general combination network. In an $(n, m)$ combination network, there are $n$ edge servers, and the logical end users can access distinct subsets of $m$ edge servers. The network coding gain is equal to

$$m\left(1 - \frac{m}{n} + \frac{1}{n}\right)$$

which is approximately equal to $m$ when $n$ is large compared with $m$. As $m$ can be an arbitrarily large integer, this implies that the network coding gain is theoretically unbounded! See also [6] for a discussion.

As an example, for $m = 2$, we have seen that for $n = 3$, i.e., the (3,2) combination network, the network coding gain is 1.33. As another example, for $n = 20$ and $m = 10$ (which are parameters used in practice), the network coding gain is 5.5. The corresponding storage saving is 81.8%, which is very significant!

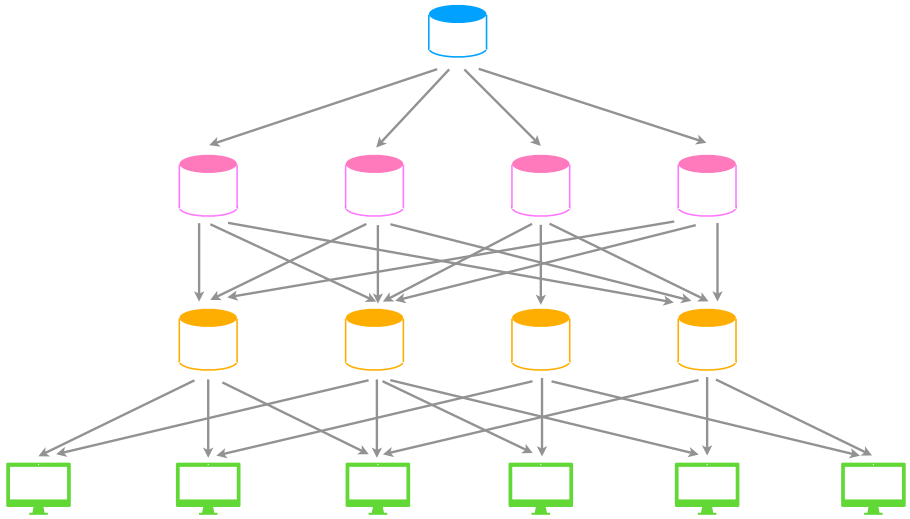


Fig. 4 A CDN with one layer of nodes between the source node and the edge servers.

In the examples we have discussed, there is one layer of CDN nodes (the edge servers) between the source node and the end users. In a general CDN, there can be additional layers of nodes between the source nodes and the edge servers. This is illustrated in Fig. 4. The idea is that the hot data (data that are frequently accessed) are stored at the edge servers to minimize the access delay, while the cold data are stored at CDN nodes that are not directly connected to the end users. In Fig. 4, the hot data are stored at the orange servers and the cold data are stored at the pink servers.

Again, in Fig. 4, a file is multicast from the source node to the end users. It is readily seen that such a network is very rich of multi-path and multi-hop from the source node to the end users. In general, the more complex the network, the higher the network coding gain. Therefore, the network coding gain of a general CDN is expected to be high.

Network coding provides the maximum flexibility without having to worry about how to route the data packets to the end user through the intermediate nodes (which is particularly problematic if the connectivity is dynamic), and at the same time achieves the maximum possible throughput.

We end this section with an example showing the advantage of network coding for storing cold data in a multi-layer CDN. In Fig. 5, $A$, $B$, $C$, and $D$ are 4 symbols representing cold data that are stored at the pink nodes. In (a), a routing solution is shown where 3 symbols $A$, $B$, and $C$ are delivered to the end users. In (b), a network coding solution is shown for which 4 symbols $A$, $B$, $C$, and $D$ are delivered to the end users. Note that in both cases exactly two symbols are stored in each of the pink nodes. When the cold data are fetched by the end user, information is transmitted to the end user via the edge nodes, but the information is not stored there. Note that in (b), network coding is performed at the two pink nodes, where the symbols $A$ and $B$ are combined into $A + B$ and on the symbols $C$ and $D$ are combined into $C + D$. For this example, the network coding gain is 4/3 and the storage saving is 25%.

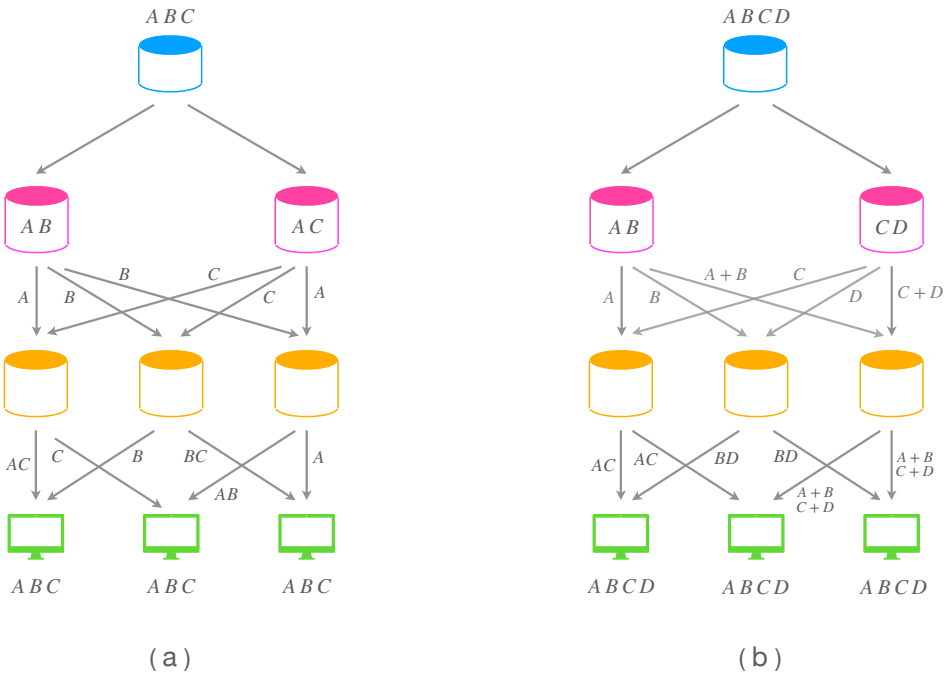



Fig. 5 A comparison between routing and network coding for a hierarchical CDN: (a) routing; (b) network coding.

# BATched Sparse Code (BATS)

For many real-world systems, the main obstacle for the application of network coding is the computation required for the coding operations. In 2011, the two main founders of n-hop, Prof. Shenghao Yang and Prof. Raymond Yeung, addressed this problem by inventing a highly efficient network coding algorithm called BATS [7][8]. The computational requirement of BATS is much lower than regular network coding, so that it can be implemented on platforms with very low computing power, e.g., IoT.

BATS consists of an outer code and an inner code. The outer code is a matrix fountain code, while the inner code is a network code. With this outer/inner code structure, encoding, recoding, and decoding of a BATS code can be done very efficiently. Moreover, BATS is provably capacity achieving. Also, since recoding at the intermediate nodes operates in the form of a pipeline, very little processing delay is incurred. These desirable features set BATS apart from all other network coding implementations.

BATS is an excellent candidate for network coding in distributed data storage. In particular, if the distribution storage system is deployed over a wide-area network where there is packet loss on the communication links, the robustness and efficiency of BATS is unmatched by any other solution.

# Network Coding Accelerator

DPU is a relatively new class of programmable computer processor designed for offloading and accelerating data-centric tasks in data centers and network infrastructure. These tasks may include packet parsing, routing, and filtering. A DPU can be used as a stand-alone processor, but it is more often incorporated into a SmartNIC, a specialized network interface controller card. Generally speaking, SmartNICs with DPU can provide advanced networking features, including hardware acceleration for virtualization, load balancing, and traffic shaping.

DPU is used as a critical component in a next-generation server. The main applications of DPUs are in data centers, where a humongous amount of data is processed and moved within and between data centers. According to NVIDIA, together with CPU and GPU, DPU will form the three pillars of computing.

Network coding can help DPU to deliver high throughput and low latency communications. Based on the discussions in the earlier sections regarding the benefits of network coding in distributed storage, we propose a next-generation DPU that is NC-integrated (NC stands for network coding). This can be achieved by incorporating an NC accelerator in the DPU. The NC accelerator consists of a highly optimized parallel hardware for network coding computation, and it supports different types of network coding algorithms, in particular BATS.

We aim to embed the NC accelerator in a DPU semiconductor chip. The NC accelerator will become an added feature of the enhanced DPU chip, making it more advanced and sophisticated than existing DPU products and more well-suited to meet future network demands.

The NC accelerator can be built in the form of a soft IP core for semiconductor. The NC accelerator soft IP core assumes a mature design that does not need further changes. A DPU semiconductor chip with this NC accelerator soft IP core embedded should be the most cost-effective and stable product solution. It is therefore an important strategy to develop a DPU chip with the NC accelerator integrated to rapidly increase market share and ultimately secure the largest market share.

Alternatively, the NC accelerator can be implemented in the form of a standalone Application-Specific Integrated Circuit (ASIC). Moreover, it can be adopted in a network interface card (NIC). The NIC can be plugged into any server to make it network coding ready. Compared with integrating the NC accelerator soft IP core into the DPU, the ASIC solution provides flexibility for deployment for the legacy data center server and data storage markets.

In summary,

- Network coding is a great technology that can address network transport and content storage issues in distributed data storage, such as CDN;
- Given the resource requirements of a CDN, DPU implementation of network coding is desirable since it offers very significant reduction in CAPEX and OPEX;
- Reduction in CAPEX is achieved by reducing the CPU and storage requirements for the CDN nodes;
- Reduction in OPEX is achieved by reducing the bandwidth requirements for connectivity and power consumption at the CDN nodes.
- BATS is the most efficient network coding algorithm and is an excellent candidate for network coding in distributed data storage.

# References


[1] R. W. Yeung, “Multilevel diversity coding with distortion,” IEEE Transactions on Information Theory, vol. IT-41, pp. 412-422, Mar 1995.

[2] R. W. Yeung and Z. Zhang, “Distributed source coding for satellite communications,” IEEE Transactions on Information Theory, vol. 45, no. 4, pp. 1111-1120, May 1999.

[3] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, “Network information flow,” IEEE Transactions on Information Theory, vol. 46, no. 4, pp. 1204-1216, 2000.

[4] S.-Y. R. Li, R. W. Yeung, and N. Cai, “Linear network coding,” IEEE Transactions on Information Theory, vol. 49, no. 2, pp. 371-381, Feb 2003.

[5] C. K. Ngai and R. W. Yeung, “Network coding gain of combination networks,” 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[6] J. Cannons, R. Dougherty, C. Freiling, and K. Zeger, “Network routing capacity,” IEEE Transactions on Information Theory, vol. 52, no. 3, pp. 777-788, Mar 2006.

[7] S. Yang and R. W. Yeung. “Coding for a network coded fountain,” 2011 IEEE Communication Theory Workshop, Sitges, Catalonia, Spain, Jun 20-22, 2011.

[8] S. Yang and R. W. Yeung, BATS Codes: Theory and Practice, Morgan & Claypool Publishers, 2017.