

Avalanche: A Network Coding Analysis

Raymond W. Yeung
Department of Information Engineering
The Chinese University of Hong Kong
N.T., Hong Kong
Email: whyeung@ie.cuhk.edu.hk

Abstract—Recently, there has been a lot of discussions on Avalanche, a Microsoft prototype for large scale content distribution on a peer-to-peer network that uses network coding as the core technology. In this paper, we present an analysis of Avalanche, showing that it achieves the theoretical lower bound on the file download time.

I. INTRODUCTION

Microsoft has recently announced a prototype called Avalanche for large scale content distribution on peer-to-peer networks that uses network coding [1] the core technology [8]. This prototype can be regarded as Microsoft's version of BitTorrent [2]. For quite some time, BitTorrent and other peer-to-peer applications (e.g., eDonkey [3] and BitComet [4]) have dominated the Internet traffic [5] [6]. What distinguishes Avalanche from BitTorrent is the application of network coding in the former. It has been demonstrated through simulation of scenarios of practical interest [7] that Avalanche can improve the expected file download time over BitTorrent by 20 to 30%. Potentially, Avalanche can be incorporated in Windows Vista [9], so that hundreds of millions of PCs on the Internet will form a gigantic peer-to-peer network. On such a network, software updates, patches, entertainment contents, etc, can be delivered efficiently to a large number of users in a scalable manner. In this paper, we analyze of the performance of Avalanche from the network coding perspective.

The rest of the paper is organized as follows. In Section 2, we describe how Avalanche works. Then in Section 3, we present a network coding analysis of Avalanche. Concluding remarks are in Section 4.

II. HOW AVALANCHE WORKS

In this section, we give a brief introduction to Avalanche. We will only focus on those aspects of the system which are relevant to our analysis. For further details on Avalanche, we refer the reader to [7].

Consider the distribution of a file originally residing on a single server to a large number of users by means of a so-called *collaborative content distribution network*. Examples are BitTorrent and Avalanche. In such a system, the server does not upload the file to each individual user. Rather, it divides the file into k data blocks, B_1, B_2, \dots, B_k , and uploads possibly coded versions of these blocks to different users at random. These users again help distributing the file by uploading blocks to other users in the network. By means of such repeated operations, information can be dispersed very rapidly, and the file is eventually delivered to every user in the network.

In a collaborative content distribution network, new users can join the network as a node at any time as long as the distribution process is active. Upon arrival, the new user will contact a *tracker* (a centralized server) that provides a subset of other users already in the system,

forming the set of neighboring nodes of the new user. Subsequent information flow in the network is possible only between neighboring nodes.

What distinguishes Avalanche from BitTorrent or any other collaborative content distribution network is the application of network coding in the former, which was introduced in [1]. Compared with routing, network coding allows coding at the nodes within the network, and it offers the benefit of achieving the maximum possible throughput when information is multicast in a point-to-point network. It was further shown in [10] [11] that linear network coding suffices to achieve the maximum throughput when a single message (e.g., a file) is to be multicast, which is the scenario under consideration. For a tutorial on network coding, we refer the reader to [12].

In Avalanche, the data blocks B_1, B_2, \dots, B_k are represented by symbols in a large finite field F , referred to as the *base field*, which has size of the order 2^{16} . At the beginning of the distribution process, a Client A contacts the server and obtains a number of *encoded blocks*. For example, the server uploads two encoded blocks E_1 and E_2 to Client A, where for $i = 1, 2$,

$$E_i = c_1^i B_1 + c_2^i B_2 + \dots + c_k^i B_k, \quad (1)$$

with c_j^i , $1 \leq j \leq k$ being chosen randomly from the base field F . Note that each E_1 and E_2 is some random linear combination of B_1, B_2, \dots, B_k . In other words, instead of choosing two particular uncoded data blocks to upload, the server forms two encoded blocks by applying network coding to all the blocks it possesses and uploads them to Client A.

Associated with each encoded block E_i ($i \geq 1$) is a *coefficient vector* \mathbf{u}^i that gives the necessary information to construct E_i from B_1, B_2, \dots, B_k . We have explained how the blocks E_1 and E_2 are formed, and it is readily

seen from (1) that for $i = 1, 2$, $\mathbf{u}^i = [c_j^i]$. When the block E_i is uploaded, the coefficient vector \mathbf{u}^i is attached.

In general, whenever a node needs to upload an encoded block to a neighboring node, the block is formed by taking a random linear combination of all the blocks possessed by that node. Continuing with the above example, when Client A needs to upload an encoded block E_3 to a neighboring Client B, we have

$$E_3 = c_1^3 E_1 + c_2^3 E_2, \quad (2)$$

where c_1^3 and c_2^3 are randomly chosen from F . Substituting (1) into (2), we obtain

$$\begin{aligned} E_3 &= \sum_{j=1}^k (c_1^3 \cdot c_j^1 + c_2^3 \cdot c_j^2) B_j \\ &= \sum_{j=1}^k (c_1^3 \cdot u_j^1 + c_2^3 \cdot u_j^2) B_j. \end{aligned}$$

Thus the coefficient vector \mathbf{u}^3 is given by $c_1^3 \cdot \mathbf{u}^1 + c_2^3 \cdot \mathbf{u}^2$. Together with \mathbf{u}^3 , the block E_3 is uploaded to Client B.

The exact strategy for downloading encoded blocks from neighboring nodes so as to avoid receiving redundant information depends on the implementation, and two such strategies were proposed in [7]. The main idea is that downloading from a neighboring node is necessary only if the neighboring node has at least one block not in the linear span of all the blocks possessed by that particular node. Upon receiving enough linearly independent encoded blocks, a node is able to decode the whole file.

Intuitively, the application of network coding can reduce the file download time because a coded block uploaded by a node contains information about every block possessed by that node. Moreover, if a node leaves the network before the end of the distribution process, it is more likely that the remaining nodes have all the information necessary for recovering the whole file when network coding is used. These

have been discussed in [7]. In the next section, we give a quantitative analysis to substantiate these claimed advantages of using network coding.

III. MODEL AND ANALYSIS

In real operation of Avalanche, blocks of data are transmitted between neighboring nodes in an asynchronous manner, and possibly at different speeds. To simplify the analysis, we assume that every transmission from one node to another neighboring node is completed in an integer number of time units. We therefore unfold the graph G in discrete time into a graph $G^* = (V^*, E^*)$ with the node set

$$V^* = \{(i, t) : i \in V \text{ and } t \geq 0\},$$

where the node $(i, t) \in V^*$ corresponds to the node $i \in V$ at time t . The edge set E^* , specified in the following, is determined by the strategy adopted for the centralized server as well as for the nodes in V to request uploading of data blocks from neighboring nodes. There is an edge with capacity m from node (i, t) to node (j, t') , where $t < t'$, if m blocks are transmitted from node i to node j , starting at time t and ending at time t' . For each $i \in V$ and $t \geq 0$, there is an edge with capacity k from node (i, t) to node $(i, t + 1)$. Such an edge models that the blocks generated/received at a node are retained in that node indefinitely over time. Without loss of generality, we may assume that all the blocks received by nodes $(i, l), l \leq t$ are transmitted on the edge from (i, t) to $(i, t + 1)$. An illustration of the graph G^* up to $t = 3$ is given in Figure 1, where the edges with infinite capacities are lightened for clarity. Note that the graph G^* is acyclic regardless of whether the graph G is acyclic or not.

We are now ready to determine the time it takes for a particular node $i \in V$ to receive the whole file. Consider the graph G^* and regard

node $(s, 0)$ as the source node multicasting the whole file with k data blocks to all the other nodes in G^* via random linear network coding. Denote the maximum flow from $(s, 0)$ to a node $v \in G^*$ by $\text{maxflow}(v)$. When the base field is sufficiently large, those nodes (i, t) such that $\text{maxflow}((i, t)) \geq k$ can receive the whole file with probability close to 1 [13]. In other words, with high probability, the time it takes a node $i \in V$ to receive the whole file is precisely the minimum t such that $\text{maxflow}((i, t))$ is at least k . Obviously, this is a lower bound on the time it takes a node $i \in V$ to receive the whole file, and it is achievable by the system under investigation with high probability. In the rare event that node i cannot decode at the minimum time t such that $\text{maxflow}((i, t))$ is at least k , it can eventually decode upon downloading some additional encoded blocks from the neighboring nodes.

When some nodes leave the system before the end of the distribution process, an important question to ask is whether the remaining nodes have all the information necessary for recovering the whole file. This question is again easy to answer when network coding is applied as prescribed. To be specific, assume that a subset of users $U^c \subset V$ leave the system after time t , and we want to know whether the users in $U = V \setminus U^c$ have the information necessary to recover the whole file, and if they do, by further exchanging information among themselves, every user in U can eventually receive the whole file (provided that no more nodes leave the system). Toward this end, consider the graph G^* . Let $U(t) = \{(u, t) : u \in U\}$, and denote the maximum flow from the node $(s, 0)$ to the set of nodes $U(t)$ by $\text{maxflow}(U(t))$. If $\text{maxflow}(U(t)) \geq k$, then the users in U have the information necessary for recovering the whole file with high probability (when the base field is sufficiently large). This is almost the best possible performance one can expect from

such a system, because if $\text{maxflow}(U(t)) < k$, it is simply impossible for the users in U to recover the whole file even if they are allowed to exchange information among themselves.

IV. CONCLUSION

In this paper, we have presented a network coding analysis of Avalanche, showing that it achieves the theoretical lower bound on the file download time with respect to the strategy adopted for the centralized server as well as for the nodes in the network to request uploading of data blocks from neighboring nodes. However, the actual performance of the system is affected by how the strategy is chosen, which involves consideration of computational efficiency, incentive mechanisms, etc. We refer the reader to [7] for further details. Further research along this line may include performance optimization of the system based on the analytical framework presented in this paper.

ACKNOWLEDGMENT

The author thanks Alan Lam for his useful inputs.

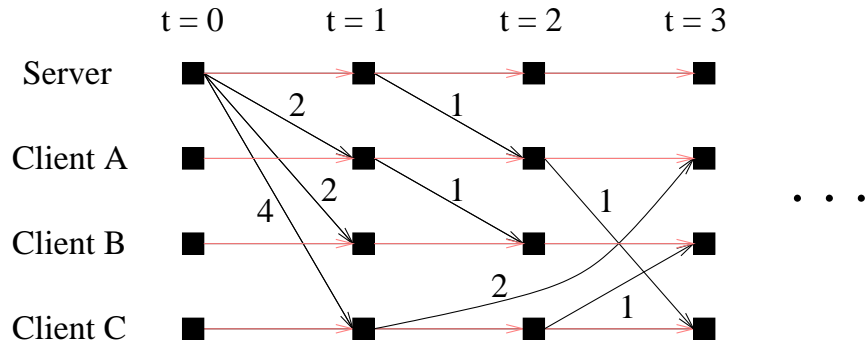


Fig. 1. A illustration of the graph G^*

REFERENCES

- [1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, IT-46: 1204-1216, 2000.
- [2] B. Cohen, "Incentive build robustness in BitTorrent," in *P2P Economics Workshop*, Berkeley, CA, 2003.
- [3] eDonkey, <http://www.edonkey2000.com>
- [4] BitComet, <http://www.bitcomet.com>
- [5] NetworkWorld, <http://www.networkworld.com/news/2005/082905-p2p.html>
- [6] Linux Reviews, http://linuxreviews.org/news/2004/11/05_p2p/
- [7] C. Gkantsidis and P. Rodriguez, "Network Coding for Large Scale Content Distribution," *INFOCOM 2005*, Miami, FL, Mar 13-17, 2005.
- [8] BBC, <http://news.bbc.co.uk/1/hi/technology/4110302.stm>
- [9] Microsoft Windows Vista, <http://www.microsoft.com/windowsvista>
- [10] S.-Y. R. Li, R. W. Yeung and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, IT-49: 371-381, 2003.
- [11] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on network coding*, vol. 11, 782-795, 2003.
- [12] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang, "Theory of network coding," submitted to *Foundations and Trends in Communications and Information Theory*. Preprint available at <http://iest2.ie.cuhk.edu.hk/~whyeung/post/netcode/main.pdf>
- [13] T. Ho, R. Koetter, M. Médard, D. R. Karger and M. Effros, "The benefits of coding over routing in a randomized setting," 2003 IEEE International Symposium on Information Theory, Yokohama, Japan, Jun 29-Jul 4, 2003.