A Layered Multi-Tree IP Multicast Protocol With Network Coding

Yining Li, Xuelei Tan, Hui Li
Shenzhen Key Lab of Cloud Computing
Shenzhen Graduate School, Peking University
Shenzhen, China.
liyining@sz.pku.edu.cn, tanxl@sz.pku.edu.cn,
lih64@pkusz.edu.cn

Jinguo Quan
Shenzhen Graduate School, Tsinghua University
Shenzhen, China.
quanjg@sz.tsinghua.edu.cn

Abstract—**This paper discusses the scenario of multi-rate multicasting to heterogeneous receivers. We adopt the Multi-Resolution Code as the layered source coding scheme, and proposed Forest, a layered multicast protocol based on multiple distribution trees. Each tree transmits a multicast layer, and Network Coding is allowed between different multicast layers. Compared to the existing solutions, our approach is completely distributed and with high performance and low complexity. Also, our approach provides a receiver-driven service model, as well as a complete group management that supports dynamic joins/leaves. These features are vital to the feasibility of a practical deployment. Simulation shows that performance and feasibility are well balanced in our approach.**

Keywords-layered multicast; network coding; multi-tree

I. INTRODUCTION

Multicast is the resource-saving way to transmit streaming media to multiple receivers. But different receivers have different capabilities and requirements. A single-rate multicast flow could either overwhelm the low-capacity receivers, or starve the high-capacity receivers [1]. Multi-rate multicasting has gained more attention since 1990's when single-rate multicasting was found insufficient to fulfill the conflicting requirement of a set of heterogeneous receivers. Today, the heterogeneity and scale of the Internet are growing explosively; so does the proportion of the Internet traffic consumed by streaming media applications. It is desirable to provide each receiver a rate that can commensurate with receiver's capability and requirement [2]. One instinct way to do multi-rate multicast is to split the original stream into layers, then transmit each layer of the original stream on an independent single-rate multicast sub-session [1]. Receivers adjust their number of subscribed sub-sessions according to their own demand.

Network coding (NC) is a promising paradigm in the field of information theory [3]. NC brings new features to the transmission of streaming media such as throughput gains, security and load balancing, etc. It is proven max-flow rate of a single-rate multi-cast session, which equals to the minimum value of the max-flow rate from each subscriber to the source, can be achieved by using linear network coding [4]. In the layered multi-rate scenario, NC can be applied within each layer to provide max-flow transmission layer by layer [5]. NC can also be applied across layers to provide more complete optimization.

Apart from the above, a lot of works have been done in the layered multicast direction. Eros [6] and Goyal [7], respectively, showed two mainstream layered source coding technologies that have been applied to the layered multicasting, Multi-Resolution Coding (MRC) and Multiple Description Coding (MDC). Mingkai [8] showed that if network coding (NC) is allowed, a MRC-based intra-layer NC solution always outperforms or at least performs the same as the MDC-based Uneven Erasure Protection (UEP) solution. So we adopt MRC as the source coding scheme in our approach. Kim [9] proposed a pushback algorithm to gather the requirements of the receivers before distributing data. But it did not explicitly distinguish different multicast layers. And the major disadvantage of pushback algorithm is intermediate nodes needed to perform NC decoding operations to fulfill the requirements of the receivers, which is extremely resource consuming. In our approach, intermediate node decoding is not required. Shao [10] attempted to combine linear NC with rainbow network flow and got a higher network throughput than original rainbow network. But this approach is related to the linear broadcast problem. We mainly focus on multicast. Mingkai [11] proposed an inter-layer NC approach to layered multicast that allowed NC of data in different layers. And higher throughput could be gained with the increasing of related cost. Zhao [12] proposed a heuristic algorithm to organize receivers into layered meshes. While in our approach, we use the thought of MRC in layered algorithm. Other related researches include [13].

In this paper, we propose Forest with a full name IP Multicast Forest (IPMF). It is a layered multicast protocol with NC applied. The novelty of our approach lies in the following. First, we create multiple distribution trees, one for each multicast layer. NC between distribution trees is allowed. Second, we introduce the Coding Matrix (CM) to be multicast to all potential subscribers. The CM indicates the distribution of the multicast layers, and NC layers, which is the linear mixing of some multicast layers. Before subscribing, potential subscribers are required to inquire the CM to decide which sub-sessions to join.

The rest of this paper is organized as follows: Section 2 describes the Forest framework. Section 3 discusses main implementation details of Forest. Simulation settings and results are presented in Section 4. Section 5 concludes the whole paper and introduces our future work.

II. FOREST FRAMEWORK

A. *Basic Idea*

The basic idea of Forest is to create multiple distribution trees. Forest splits and recodes the original multicast flow at the source node into sub-flows, using algorithms described in [6]. We first split out the most basic, important data, and then recode them to form a "base sub-flow", while other data "enhancement sub-flows". Each sub-flow is associated with a sub-source node and a sub-group address. Multiple multicast layers have multiple distribution trees. We call it the sub-tree for it is logically part of the original multicast distribution structure. Subscribers that want to subscribe to the original multicast session, have to collect all the sub-flows to rebuild the original flow. From this point of view, Forest is an extension to the traditional tree-like distribution structure. It inherits the convenient group member management of the distribution tree, while expanding its transmission performance.

While, this Forest structure may also encounters the so-called "Multicast Packing Problem", for sub-trees in the forest must be edge-disjoint to avoid congestion. When sub-tree collision happens, bottlenecks may emerge, and transmission performance will drop rapidly. Luckily, NC can simplify this problem. In Forest, when collision happens at a node between multicast layers, we can code them together.

B. *Forest Structure*

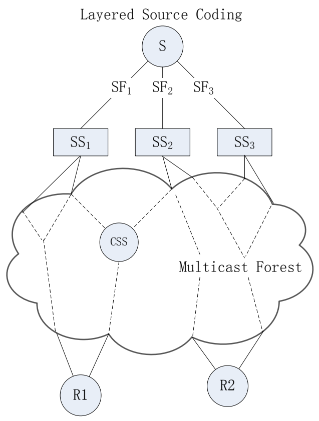

Figure 1. Main Forest structure

Figure 1 shows the main Forest structure. There are 5 different kinds of nodes in the Forest architecture: source, sub-source, coding node, forwarding node and receiver. The entire Forest architecture can be partitioned into two parts: layered source coding, and multicast forest. And some main definitions of terms as well as their features used in this paper are showed below.

In Figure 1, S and R, respectively, represent the source and receiver.

SF: A sub-flow (SF) is a data flow, formed by splitting and/or recoding the original multicast flow at the source node.

SS: A sub-source (SS) node is a different node. One sub-source is assigned to one sub-flow. A sub-source node will receive and cache sub-flow information from the source node. And it will act as a new source including constructing the tree-like multicast distribution structure, managing group members, and sending data out to the output interface list.

ST: A sub-tree (ST) is the multicast distribution tree constructed by a sub-source node, using IGMP join/prune messages, and is consisted of intermediate nodes and links. All the sub-trees that belong to the original multicast session form a so-called coding-sub-graph of IPMF.

Multicast Forest: In Figure 1, if we cover the flow-splitting part, the multicast forest part can be seen as multiple independent multicast distribution trees, each transmitting a sub-flow to the same receiver group concurrently.

CSS: A node who begins to carry out coding operations when collision happens, will automatically transform into a "coding-sub-source" (CSS). In most cases, a CSS acts just like other independent sub-sources, but, there are still differences. Firstly, a CSS receives information from some sub-flows not only one. Secondly, multiple CSSs may sit on one physical coding node like Figure 2. Thirdly, if a coding-sub-flow has *k* parent sub-flows, then, when a group receiver collects sub-flows, the coding-sub-flow can replace any one of its parent sub-flows, only when the "sub-flow coefficient matrix" of that receiver is reversible after the replacement. The last, when a CSS emerges, it will send an inform message to the source node to register itself, and notify the source node about its coding coefficient vector. It will also send a message through all its registered output interfaces to inform downstream nodes about the change.

AG: We use Auxiliary Group (AG) which has a fixed and public group address to multicast the following information to all group receivers and potential receivers:

a) Active groups and its sources in the current Automomous System (AS).

b) Sub-source addresses, sub-group addresses, coding-sub-source addresses and coding-sub-group addresses of a specific group.

c) Distribute sub-flow coefficient matrix.

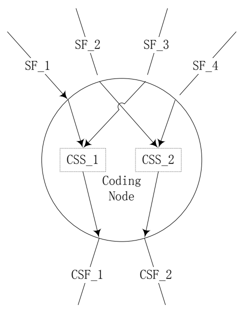

Figure 2. One physical node with two CSSs

In IPMF, we adopt a traditional Shortest Path Tree (RPT) structure as AG. Simulation results show that AG works well in doing its job.

III. Forest Implementation Details

A. Coding Strategy

IPMF uses a flow-oriented linear NC strategy. Sub-flow is the smallest unit. So coefficients are chosen for flows, not individual packets. For example, In Figure 3, if we chose α and β as coefficients for sub-flow *SF_1* and *SF_2*, then all the packets in the coding-sub-flow *CSF* can be calculated as:

$$\mathrm{Packet}_{\mathrm{CSF}} = \alpha \times \mathrm{Packet}_{\mathrm{SF}_1} + \beta \times \mathrm{Packet}_{\mathrm{SF}_2} \qquad (1)$$

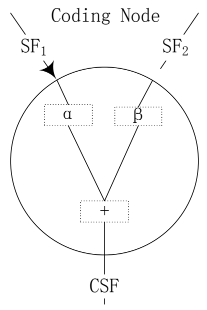

Figure 3. Flow-oriented NC

The advantages of flow-oriented NC coding are the followings:

a) Flows are easier to manage than individual packets.

b) Provide basis for routing, and decoding.

c) Reduce the size of Galois Field (GF), because less random coefficients are needed.

d) Implement "user-driven" content delivery effectively.

In IPMF, each SF including CSF is associated with a coefficient vector (CV):

$$\mathrm{CV} = [\,c_1, c_2 \ldots, c_N]^{\mathrm{T}} \qquad (2)$$

where N is the number of sub-flows, excluding coding-sub-flows, and c_i, $1 \le i \le N$, is the coefficient number randomly chosen from GF (2^p). If we associate a sub-flow number from 1 to N and each sub-flow, then the CV of a sub-flow generally describes its composition. For example, in Figure 3, the CVs for SF_1, SF_2 and CSF are $\mathrm{CV}_{\mathrm{SF}_1} = [1,0,0,0]^{\mathrm{T}}$, $\mathrm{CV}_{\mathrm{SF}_2} = [0,1,0,0]^{\mathrm{T}}$ and $\mathrm{CV}_{\mathrm{CSF}} = [\alpha, \beta, 0,0]^{\mathrm{T}}$ respectively.

We use these CVs to form a coefficient matrix (CM) in the source node, then multicast the CM through auxiliary group. We assume at a certain time, there are *M* CSSs in the network.

$$CM = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & | & \alpha_{11} & \alpha_{21} & \cdots & \alpha_{M1} \\ 0 & 1 & 0 & \cdots & 0 & | & \alpha_{12} & \alpha_{22} & \cdots & \alpha_{M2} \\ 0 & 0 & 1 & \cdots & 0 & | & \alpha_{13} & \alpha_{23} & \cdots & \alpha_{M3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 & | & \alpha_{1N} & \alpha_{2N} & \cdots & \alpha_{MN} \end{pmatrix} \qquad (3)$$

The first part of the CM is a $N \times N$ unit matrix, representing N sub-flows. The second part of CM is a $N \times M$ matrix, representing M coding-sub-flows. When a group receiver collects one sub-flow, it collects one column of the CM.

When a group receiver collects N sub-flows, including CSSs, it collects a $N \times N$ matrix, called the receiving matrix (RM). If the RM is reversible, i.e., the receiver collects enough information about the original multicast flows, receivers can successfully decode and rebuild all the original information. While, due to heterogeneity, when a group receiver only collect P ($P<N$) sub-flows including coding-sub-flows, we can still decode part of the information if enough variables can be eliminated by Gaussian elimination, and if the "base sub-flow" can be decoded, then the receiver is still able to enjoy the service in a lower quality.

B. Multicast Forest Construction

1) Constructing The Forest at The Beginning

At the beginning of the multicast session, there is no CSS in the network, thus the CM initial group receivers get from the auxiliary group is simply a $N \times N$ unit matrix:

$$CM_{init} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \qquad (4)$$

Algorithm 1 can construct multicast forest. At the beginning, we need to state a few assumptions. Firstly, a node will operate NC, only when multiple incoming sub-flows share the same output interface, and the overall bandwidth of these incoming sub-flows exceeds the effective bandwidth of the output interface. Secondly, we assume all sub-flows have the same bandwidth, all physical links in the network have the same bandwidth, and two sub-flows cannot be transmitted through one link at the same time. The last, current version only supports N=2.

```
Algorithm 1
for each receiver do
```

```
    Listen to auxiliary group and update the CM.
    for each sub-flow, higher priority first do
      int $i$ = 1
      while $i$ do
      if exist unoccupied interfaces then
          Lookup the corresponding SS address in the routing table, with $i_{th}$ best path.
          if returned interface in unoccupied then
            Occupy the interface.
          else
            if $i$< number of interfaces then
                $i$+ +; continue
            else break
            end if
        else break
        end if
      end while
    end for
    for all SFs that have occupied an interface do
      Send Source-Specific-Join with the occupied interface.
    end for
end for
```

This algorithm generally assigns a different interface for each sub-flow ordered by priority with its best effort. After join messages are sent, sub-tree collision may happen. Some nodes may transform into CSS. In this case, a solution is given in the next situation.

2) Reconstructing Forest During Run-time

The multicast session has already been activated for a while. The CM is no longer a unit matrix due to sub-tree collisions, and then:

$$CM_{rec} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & | & \alpha_{11} & \alpha_{21} & \cdots & \alpha_{M1} \\ 0 & 1 & 0 & \cdots & 0 & | & \alpha_{12} & \alpha_{22} & \cdots & \alpha_{M2} \\ 0 & 0 & 1 & \cdots & 0 & | & \alpha_{13} & \alpha_{23} & \cdots & \alpha_{M3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & | & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 & | & \alpha_{1N} & \alpha_{2N} & \cdots & \alpha_{MN} \end{pmatrix} \quad (5)$$

a) Let $CV_{CSF} = [\alpha_1, \alpha_2, \alpha_3 ..., \alpha_N]^T$.

b) Let $Prio_{id}$ represents the priority of SF_{id}.

c) All the non-zero elements in CV_{CSF} form a set: $CV'_{CSF} = \{\alpha_{i_1}, \alpha_{i_2}, \alpha_{i_s}\}$, $s \le N$.

d) Then $\mathrm{Prio}_{CSF} = \frac{\sum_{j=1}^{S} \mathrm{Prio}_{i_j}}{S}$.

For example, if CV_{CSF}=[1,0,1,1,0]^T , then $\mathrm{Prio}_{CSF} = \frac{\mathrm{Prio}_1 + \mathrm{Prio}_3 + \mathrm{Prio}_4}{3}$.

Algorithm 2 reconstructs the multicast forest during run-time. This algorithm generally assigns a different interface for each sub-flow including coding-sub-flow in the reversible RM with its best effort. It also takes the priority into account.

Algorithm 2

```
for each receiver whose Join has been denied do
    Listen to auxiliary group and update the CM.
      for each SF whose Join has been denied, higher priority first do
    Pick out a CSF satisfies: The SF-id$_{th}$ element in CV is nonzero. Meaning this CSF contains information about the SF in question.
    Higher priority first.
    Send Source-Specific-Join toward the selected CSF.
    end for
end for
for each potential receiver do
    Listen to auxiliary group and update the CM.
    Pick out $N$ SFs including CSF from the CM, satisfying:
    The $N \times N$ RM constructed is reversible;
    The $N \times N$ RM constructed has the highest possible priority;
    for each SF in RM, higher priority first do
      int $i$=1
      while $i$ do
      if there are still unoccupied interfaces then
          Lookup the corresponding SS address in the routing table with $i_{th}$ best path.
          if returned interface is unoccupied then
            Occupy the interface.
          else if   $i$< number of interfaces then
                $i$ ++; continue
          else if exist SFs unselected in CM then
              Replace this SF, so that RM satisfies:
    The $N \times N$ RM constructed is reversible;
    The $N \times N$ RM constructed has the highest possible priority;
              else break
              end if
          end if
          end if
    else break
    end if
end while
end for
    for all SFs that have occupied an interface do
      Send Source-Specific-Join with the occupied interface.
    end for
end for
```

IV. SIMULATION

In this section, we present some simulation results obtained by NS2 network simulator software. In order to verify our core thoughts with low complexity, we chose the classical and simple "Butterfly Network" to run IPMF like Figure 4. IPMF emphasizes on multi-tree construction, so we choose the traditional single-tree structure provided by Protocol Independent Multicast-Sparse Mode (PIM-SM) as a comparison object.

In our simulation, there are three key aims with IPMF. Firstly, we want to test and verify the dynamic multicast forest construction algorithm of IPMF. Secondly, we want to test and verify the support for dynamic joins/prunes. The last, we want do throughput test compared with PIM-SM.

Here are details of simulation settings with NS2. The original flow is split into two SFs; node 1 and 2 are SS nodes; node 3 is a CSS node; node 5 and 6 are group receivers; all links have a bandwidth of 1Mbps; source sending rate varies from 100 Kbps to 2.4 Mbps; as to the simulation time, node 5 joins the group at 0.0s, while node 6 joins the group at 0.3s.

Here are our Simulation results with NS2:

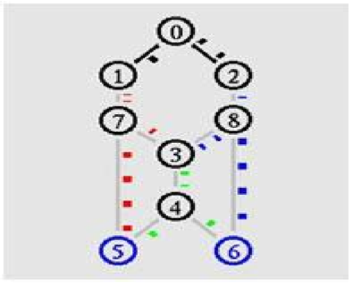

Figure 4. IPMF is well constructed in NS2

In Figure 4, at simulation time 0.3s, node 6 correctly joins the group, so we can know that the Forest multicast forest is well constructed.

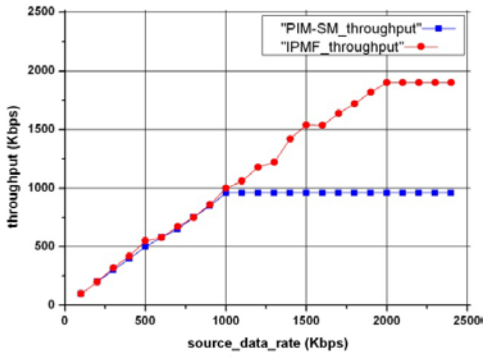

Figure 5. Throughput test

In Figure 5, the max-flow of node 5 is 2Mbps. As we change the source sending rate from 100 Kbps to 2.4 Mbps, IPMF obviously outperformed PIM-SM after the source sending rate exceeds 1Mbps.

V. Conclusions and Future Work

In this paper, we proposed Forest, a layered multi-path IP multicast protocol with network coding applied. Forest seeks the balance between performance and feasibility. We also give the IPMF construction algorithm aiming to construct a practical distribution structure and make it possible to apply network coding in a dynamic environment. Though this algorithm may fail from time to time because of its bottom-up manner, it will converge to a stable out-come by re-run. Theoretically, the more sub-flows the original multicast flow split, the better Forest performs. But more sub-flows requires more sophisticated algorithm. Simulation results show that dynamic joins/prunes are well supported by Forest. Besides, in certain situations, compared to the traditional PIM-SM protocol, Forest achieves a throughput gain up to 50%, with only two sub-flows enabled.

The current version of Forest is basically an experimental version. Optimizations will be done in the future. We will choose NS3 which has some new features compared with NS2 to implement a better and more detailed simulation. After that, we will plan to implement Forest in an embedded system to test its performance in a real network environment.

Acknowledgment

The authors would like to thank the support of National Basic Research Program of China No. 2012CB315904, NSFC No. 61179028, SZ JC201005260234A, SZ JC201104210120A, ZD201006110044A, and GDNSF No. S2011010000923.

References

[1] S. McCanne, V. Jacobson, and M. Vetterli, “Receiver-driven layered multicast,” SIGCOMM Comput. Commun. Rev, vol. 26, no. 4, pp. 117-130, 1996.

[2] B. Li and J. Liu, “Multirate video multicast over the internet: an overview,” Network, IEEE, vol. 17, no. 1, pp. 24 -29, 2003.

[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, Q. T. Sun, and Z. Shao, “Linear network coding: Theory and algorithms,” Proceedings of the IEEE, vol. 99, no. 3, pp. 372-387, 2011.

[5] N. Sundaram, P. Ramanathan, and S. Banerjee, “Multirate media stream using network coding,” in Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing, 2005.

[6] M. Eros, “Universal multiresolution source codes,” IEEE Transactions on Information Theory, vol. 47, no. 6, pp. 2113 -2129, 2001.

[7] V. K. Goyal, “Multiple description coding: compression meets the network,” Signal Processing Magazine, IEEE, vol. 18, no. 5, pp. 74 -93, 2001.

[8] S. Mingkai, “Scalable Multimedia Communication Using Network Coding,” PhD thesis, McMaster University, 2011.

[9] M. Kim, D. Lucani, X. Shi, F. Zhao, and M. Medard, “Network coding for multiresolution multicast,” INFOCOM, Proceedings IEEE, pp. 1-9, 2010.

[10] M. Shao, X. Wu, and N. Sarshar, “Rainbow network flow with network coding,” Network Coding, Theory and Application, pp. 1-6, 2008.

[11] S. Mingkai, S. Dumitrescu, and W. Xiaolin, “Layered multicast with inter-layer network coding for multimedia streaming,” IEEE Transactions on Multimedia, vol. 13, no. 2, pp. 353-365, 2011.

[12] J. Zhao, F. Yang, Q. Zhang, Z. Zhang, and F. Zhang, “Lion: Layered overlay multicast with network coding,” IEEE Transactions on Multimedia, vol. 8, no. 5, pp. 1021-1032, 2006.

[13] Z. Kiraly and E. R. Kovacs, “A network coding algorithm for multi-layered video streaming,” Network Coding, pp. 1-7, 2011.