Layer Allocation Algorithms in Layered Peer-to-Peer Streaming

Yajie Liu, Wenhua Dou, and Zhifeng Liu

School of Computer Science,
National University of Defense Technology, ChangSha, 410073, China
liu_lyj@hotmail.com

Abstract. In layered peer-to-peer streaming, as the bandwidth and data availability (number of layers received) of each peer node are constrained and heterogeneous, the media server can still become overloaded, and the streaming qualities of the receiving nodes may also be constrained for the limited number of supplying peers. This paper identifies these issues and presents two layer allocation algorithms, which aim at the two scenarios of the available bandwidth between the media server and the receiving node, to reach either or two of the goals: 1) minimize the resource consumption of the media server, 2) maximize the streaming qualities of the receiving nodes. Simulation results demonstrate the efficiencies of the proposed algorithms.

1 Introductions

Peer-to-peer streaming is cost-effective for it can capitalize the resources of peer nodes to provide service to other receivers. In general, there exist three properties on peers: 1) peer's outbound bandwidth or the bandwidth it willing to contribute is limited, there always need multiple supplying peers cooperate to serve a requesting node, 2) different peer nodes can receive and process different levels of streaming qualities, which means their data availabilities as supplying peers are also heterogeneous and constrained, 3) peers' inbound and outbound bandwidths are also heterogeneous. Layered peer-to-peer streaming has the potential to address the issues of heterogeneities, but under it there also exist some factors such as the available outbound bandwidths of peers are less than their inbound bandwidths, the number of supplying peers is limited to the receiver sometimes, the data availabilities(number of layers received) are constrained and so on, these factors can lead two results, 1) the media streaming server can still become overloaded when the system is in large scale, 2) the receiver's streaming quality may also be constrained with limited heterogeneous supplying nodes.

We identify these issues and present a layer allocation framework which contains two layer allocation algorithms. Concerning the available bandwidth between the media server and a receiver, there exist two scenarios, 1) sufficient available bandwidth, which means all the expected layers of the receiver can be solely provided by the media server, and a layer allocation algorithm is presented to minimize the resource consumption of the media server while satisfying the streaming quality of the receiver, 2) insufficient available bandwidth,

H. Jin et al. (Eds.): NPC 2004, LNCS 3222, pp. 167–174, 2004.

which means the receiver's expected layers can not solely be provided by the media server, and another layer allocation algorithm is presented to maximize the streaming quality of the receiver. Either of the algorithms is executed on the receiver side, and we only focus on the situation that a receiver has one or more candidate supplying peer nodes besides the media server, otherwise the receiver's streaming quality will only be determined by the available bandwidth between the server and the receiver, here we don't discuss it for the simplicity. In addition, we suppose that each peer node can cache all the layers that it has received and each receiving node can identify the available bandwidths between it and the candidate supplier nodes, and these available bandwidths keep unchanged without the failures of the supplier nodes.

There has been related work on layered peer-to-peer streaming. For example, [1] proposed a framework for quality adaptive playback of layered media streaming, the data allocation granularity of it was based on data packets, and ours is based on layers. To our knowledge, the most closely related work is [2], and the main differences between it and this paper include, 1) this paper presents an improved layer allocation algorithmin compares to [2], in the scenario of sufficient and with the assumption that the layer rates are heterogeneous; 2) in addition, this paper discusses and formalizes the problem in the scenario of insufficient which is not mentioned in [2], and present a heuristic layer allocation algorithm.

2 System Model and Layer Allocation Algorithms

Let p denote a receiving node, p_0 denote the media server, $l_1, l_2, \ldots, l_l$ denote the layers of the stream that p will request, with l_1 as the base layer and others as the enhancement layers, these layers are accumulative, i.e., l_i can be only decoded if layers l_1 through l_{i-1} are available. Let r_i denote the streaming rate of l_i and the layer rates are heterogeneous, $L = \{l_1, l_2, \ldots, l_m\}$ $(m \leq l)$ denote the expected layer set that p expects to receive according to its inbound bandwidth and the layer rates. Let $P = \{p_1, p_2, \ldots, p_n\}$ denote the candidate supplying peer node set of p, $b(p_j, p)$ $(0 \leq j \leq n)$ denote the available bandwidth between p_j and p, a_i denote the number of layers that p_i has cached, $x_{i,j}$ denote whether l_i will be provided by p_j, if will then assigned to 1 otherwise assigned to 0. Let $L(p_j)$ denote a subset of L where the layers are provided by p_j.

2.1 The Scenario of Sufficient

In this scenario, the goal of the layer allocation problem is to maximally save the resource consumption of the server. We can formalize this problem as follows:

$$Maximiz \sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j} * r_j \tag{1}$$

$$Subject\ to \sum_{j=0}^{n} x_{i,j} = 1, \quad i = 1, 2, \ldots, m \tag{2}$$

$$\sum_{i=1}^{m} x_{i,j} r_i \leq b(p_j, p), \quad j = 1, 2, \ldots, n \tag{3}$$

$$x_{i,j} * i \leq a_j \ . \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n \tag{4}$$

Formulation (1) indicates that the goal is to maximally allocate the expected layers to the peer set P. Equation (2) indicates each layer can only be allocated to a candidate supplier. Inequation (3) indicates the sum rate of the layers that allocated to a peer node can not exceed the available bandwidth between the peer node and p. Inequality (4) indicates the layers that a peer can provide can not exceed what it has cached.

From the formulations (1)-(4) we can see that this optimal allocation problem belongs to the integer programming (IP) problems, it is NP-Hard for these constraints especially each variable $x_{i,j}$ must be 0 or 1. Here we present an approximation algorithm for this problem which includes three steps: 1) relax the integrality of each $x_{i,j}$, and thus make the IP problem become a linear programming (LP) problem, 2) solve the LP problem by a maximum flow algorithm on a constructed directed graph, 3) adjust the flow value on the graph until each $x_{i,j}$ becomes 0 or 1. Next we will describe this approximation algorithm in details.

We first relax the integrality of each variable $x_{i,j}$ and make this optimization problem become a LP problem. As the dimension of the matrix that corresponding to the LP problem's constraints is large, here we don't try to solve it using the traditional simplex or dual-simplex algorithms, but convert it to a maximum flow problem on a directed graph $G(V, E)$. $G(V, E)$ is constructed as follows: the vertex set $V = L \cup P \cup \{s, t\}$ where s is the source, t is the sink, L and P keep the meanings mentioned above; for each l_i, direct an arc from s to l_i, its capacity is assigned r_i; for each l_i and each p_j, if p_j has cached l_i and $b(p_j, p)$ is not less than r_i, then direct an arc $e(l_i, p_j)$ with capacity r_i from l_i to p_j; for each p_j, direct an arc form p_j to t, its capacity is assigned $b(p_j, p)$. From $G(V, E)$'s construction, we can conclude that the maximum sum rate of the layers that allocated to the peers in P can not exceed the maximum flow value through $G(V, E)$.

Definition 1. *To any arc $e(l_i, p_j)$ that belongs to $\{e(l_i, p_j) | 1 \leq i \leq m, \ 1 \leq j \leq n\}$ of $G(V, E)$, if the flow value on $e(l_i, p_j)$ equals to r_i or 0, then call this arc an integral arc, otherwise a fractional arc.*

After constructing the directed graph, the next step of the algorithm is to calculate the maximum flow on $G(V, E)$, here we use the classical algorithm *MPM* [3] to calculate the maximum flow and let $w(l_i, p_j)$ denote the flow value on $e(l_i, p_j)$. After this calculation each variable $x_{i,j}$ is assigned to $w(l_i, p_j)/r_i$. If $x_{i,j}$ equals to 1, then add l_i to the set $L(p_j)$. If each arc of $\{e(l_i, p_j) | 1 \leq i \leq m, \ 1 \leq j \leq n\}$ is integral at this time, then end this algorithm for it has got an optimal allocation result; otherwise further adjust the flow on $G(V, E)$ as follows.

Construct a edge-induced subgraph $G(V', E')$ of $G(V, E)$, $G(V', E')$ is induced by those fractional arcs from the arc set $\{e(l_i, p_j) | 1 \leq i \leq m, \ 1 \leq j \leq n\}$ of $G(V, E)$, all parameters on these arcs are kept unchanged in the construction.

Definition 2. *To any vertex that belonging to P of $G(V', E')$, if a vertex's in-degree equals to 1, then call this vertex a singleton vertex.*

Repeat the following substeps until all the arcs of $G(V', E')$ are removed: if there does not exist any singleton vertex, then adjust the flow according to the rounding rule 1, else adjust the flow according to the rounding rule 2.

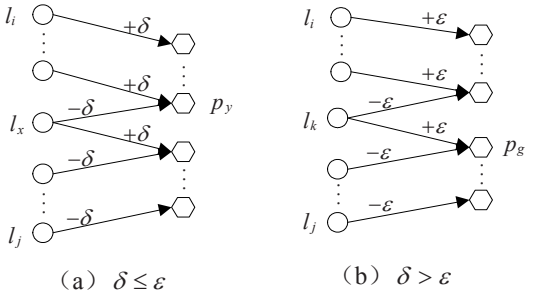


Fig. 1. The flow adjustment on $G(V', E')$

Rounding rule 1: Ignore the edge directions of $G(V', E')$ and find a longest path on $G(V', E')$. As at this time there does not exist any singleton vertex, the two end vertices of the longest path must belong to L, otherwise leads a contradiction. Suppose the two end vertices of the longest path are l_i and l_j, denote the longest path as $l_i \sim l_j$. Let δ denote the minimum flow value of all the arcs comprised in $l_i \sim l_j$, $e(l_x, p_y)$ denote the arc whose flow value equals to δ on $l_i \sim l_j$. Let ε denote the minimum remaining capacity of all the arcs comprised in $l_i \sim l_j$, and $e(l_k, p_g)$ denote the arc whose remaining capacity equals to ε on $l_i \sim l_j$. Execute either the following branches on $l_i \sim l_j$: (i) If $\delta \le \varepsilon$, adjust the flow as illustrated in Fig.1(a), we can see that the flow value on the arc $e(l_x, p_y)$ will become zero, which means at least an arc $(e(l_x, p_y))$ will become an integral arc after this type adjustment; (ii) if $\delta > \varepsilon$, adjust the flow as illustrated in Fig.1(b), we can see that the flow value on the arc $e(l_k, p_g)$ will equal to its capacity, which means at least an arc $(e(l_k, p_g))$ will become an integral arc after this type adjustment. At last remove some arcs of $G(V', E')$ as follows: to each arc (suppose $e(l_u, p_v)$) of the path $l_i \sim l_j$, if its flow value equals to zero, then remove it from $G(V', E')$; if its flow value equals to its capacity, then add l_u to $L(p_v)$, remove all the arcs that associated with l_u. We can see after the above adjustments, the flow value through each $p_j (1 \le j \le n)$ keeps unchanged, the flow value on each arc does not exceed its capacity.

Rounding rule 2: Select a existent singleton vertex on $G(V', E')$, suppose it is p_j. p_j's single neighbor vertex must belong to L, suppose it is l_i. Remove all arcs that associated with l_i, add l_i to $L(p_j)$, construct a subset of $L(p_j)$ that the layer's sum rate of this subset does not exceed the available bandwidth $b(p_j, p)$ but is nearest to $b(p_j, p)$, substitute this subset for $L(p_j)$.

The maximum flow on $G(V, E)$ can be computed in $0(|V|^3)$ time. At least an arc will be removed in either the rounding rules, and there at most exist

$m * n$ arcs on $G(V', E')$, so this layer allocation algorithm is a polynomial time algorithm.

2.2 The Scenario of Insufficient

In this scenario, as the receiving node's expected layers can not solely be supplied by the media server, the goal of the layer allocation algorithm is to maximize the streaming quality of the receiving node. Considering that a layer l_i can only be decoded if layers l_1 through l_{i-1} are available, we first define a special layer subset of L before formalize the allocation goal of this scenario.

Definition 3. *Suppose S is a subset of L, we call S a prefixed-subset of L if and only if the following condition satisfied: if layer $l_i (1 \leq i \leq m)$ belongs to S, then all the layer(s) $\{l_j | 1 \leq j < i\}$ must also belong to S.*

Suppose S is a prefixed-subset of L to be constructed, the allocation goal can be formalized as

$$Maximize \quad |S| \tag{5}$$

$$Subject\ to \quad \sum_{j=0}^{n} x_{i,j} = 1, \ i : l_i \in S \tag{6}$$

$$\sum_{i:l_i \in S} x_{i,j} r_i \leq b(p_j, p), \ j = 0, 2, \ldots, n \tag{7}$$

$$x_{i,j} * i \leq a_j \ . \ i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n \tag{8}$$

Formulation (5) indicates the goal is to maximize the size of S, which means to maximize the receiver's streaming quality by optimally allocating layers to the candidate supplying node set $P \cup \{p_0\}$. Equation (6) indicates each layer in S must be allocated to a node of $P \cup \{p_0\}$. The meanings of the formulation (7) and (8) are similar to the meanings of the formulation (3) and (4).

Theorem 1. *The optimal layer allocation problem in the scenario of insufficient is NP-Hard.*

Proof: Consider a decision problem that corresponds to the optimal layer allocation problem in the scenario of insufficient. Regard any prefixed-subset $L' = \{l_1, l_2, ..., l_k\} (k \leq m)$ of L as an item set, each item l_i is associated with a size r_i; regard the candidate supplier set $P \cup \{p_0\}$ as a bin set, each bin $p_j (0 \leq j \leq n)$ is associated with a capacity $b(p_j, p)$; for the simplicity let $a_1 = a_2 = ... = a_n = m$ which means any item is allowed to be placed into any bin. Then the decision problem can be described as: if all the items of L' can be put into the bin set $P \cup \{p_0\}$ while the sum size of the item(s) that placed into each bin does not exceed the bin's capacity. As the 3-partition problem can be regarded as a special case of this decision problem and the 3-partition problem is NP-Complete [6], so the above decision problem is NP-Complete, and the corresponding optimal layer allocation problem is NP-Hard.

Here we propose an heuristic algorithm (Fig.2) for this optimization problem, its main idea is to allocate the layers from the base layer to the enhancement

layers in turn and allocate each layer to a best suitable supplier. The algorithm's complexity is $0(mn)$.

```
1.  let U ← Φ, L(p_j) ← Φ (0 ≤ j ≤ n);
2.  for i ← 1 to m do
3.      for j ← 0 to n do
4.          if a_j ≥ i and b(p_j, p) ≥ r_i then U ← U + {p_j};
5.      end for
6.      if |U| = 0 then
7.          return i − 1 and exit; /* i − 1 is the number of allocated layers */
8.      if |U| = 1 then /* suppose U equals to {p_j}*/
9.          L(p_j) ← L(p_j) + {l_i},  b(p_j, p) ← b(p_j, p) − r_i;
10.     if |U| > 1 then
11.         select the supplier(s) that with the least available bandwidth to p ;
12.         if multiple such suppliers exist, then
13.             select a supplier which cached the least number of the layers;
14.         if the selected supplier is p_j then
15.             L(p_j) ← L(p_j) + {l_i},  b(p_j, p) ← b(p_j, p) − r_i ;
16.     let U ← Φ;
17. end for
```

Fig. 2. The heuristic algorithm to maximize a receiver's streaming quality

2.3 Fault-Tolerance Mechanism

Node departures can happen frequently in peer-to-peer community. Our fault-tolerance mechanism that adapts to this can be described as follows: when a receiving node detects that a supplying node has ceased to provide the media data, it sends a request to the media server and tries to get the missing layers from it; if the receiving node's request can be satisfied by the media server, then it will get the missing layers from the media server; otherwise, on the side of the receiving node the algorithm that described in Section 2.2 will be executed to reallocate the layers to the candidate suppliers thus try to maximize the receiving node's streaming quality.

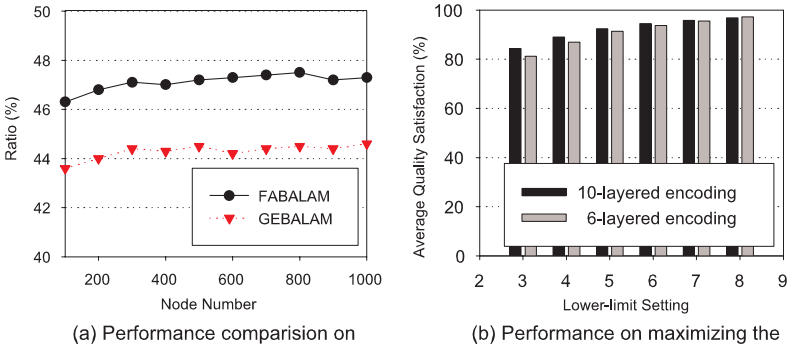
3 Simulation Results

In this section, we present simulation results to evaluate the performance of the proposed algorithms. A 6-layered encoding mode whose layer rates are from 64kbps to 192kbps and a 10-layered encoding mode whose layers rates are from 64kbps to 128kbps are used in the simulation, both with the sum rate 768kbps (denoted as R_0). The topology used in the simulation has three levels: the first two levels are generated using the GT-ITM Generator [5] which contain 400

routers, and 1000 host nodes are attached to the stub routers as the lowest level. The media server is attached to a transit router. The link bandwidths of the first level are assigned 100M, and are chosen in the range [1M, 10M] of the second level. The available inbound bandwidths of nodes are distributed in the range [128K, 1M], the outbound bandwidth that each node willing to contribute is distributed in the range $[0.1R_0, 0.6R_0]$ which reflects the diversity in the P2P community [4].

As [2] had proposed a greedy layer allocation algorithm which had the similar goal and assumptions to the algorithm presented in Section 2.1, and in the simulation we will compare the performance of the two algorithms, and name the algorithm in [2] as $GEBALAM$(GrEed Based Approximation Layer Allocation algorithM) and ours as $FABALAM$ (Flow Adjustment Based Approximation Layer Allocation algorithM) in the simulation.

A experiment is simulated as follows with first setting a lower-limit: each node joins the system by submitting a request to the server in random order; the server constructs a candidate supplying peer set P as a response to a request that received according the statuses of peers that have joined, each peer in P can at least supply a expected layer of the requesting node; if the size of P is larger than the lower-limit setting, then P and p_0 will be returned to the requesting node, and either of the algorithms presented in this paper will be executed on the requesting node side according to either the scenarios; otherwise only p_0 is returned and the requesting node will begin to get data from p_0; each node will become a peer node when it begins to receive the media data; after all nodes have joined the system, we repeat the above procedure but substitute the algorithm $GEBALAM$ for $FABALAM$. Each experiment is performed 10 times, and the simulation results are averaged over all results.



(a) Performance comparision on saving server's bandwidth consumption

(b) Performance on maximizing the streaming qualities of receiving nodes

Fig. 3. Performance of the proposed algorithms

Fig.3(a) shows the performance comparison results on saving the server's bandwidth consumption of the two algorithms with the lower-limit 6 and the 10-

layered encoding mode. The axis x denotes the number of nodes that have joined the system. The y axis denotes the ratio of the server's bandwidth consumption to the sum of the bandwidths that consumed by the joined nodes. It shows that our proposed algorithm can save more bandwidth consumption (2%∼3%) of the media server than the corresponding greedy algorithm proposed in [2].

Define a metric *streaming quality satisfaction* as the ratio of the sum rate of the layers that a node actually receives to the sum rate of its expected layers. We select these allocation samples in each experiment where the available bandwidth between a receiving node and the media server is insufficient and let the average streaming quality satisfaction of all the samples to denote the performance of the algorithm that presented in Section 2.2. From Fig.3(b) it can be seen that the streaming qualities of these receiving nodes keep at high levels, furthermore, the larger the lower-limit setting, the higher the steaming qualities, this is because the more the number of candidate supplying peer nodes, the more likely it is that a receiving node can receive more expected layers.

4 Conclusions

This paper presented a layer allocation framework for layered peer-to-peer streaming. Considering the constraints of the supplying peer nodes and the increasing trend of the server's bandwidth consumption, this framework comprised two algorithms according to the different scenarios of the available bandwidth between the media server and a receiving node to reach either or two of the goals: 1) minimize the resource consumption of the media server, 2) maximize the receiving node's streaming quality. Simulation studies show the efficiency of the layer allocation framework. Our fault-tolerance mechanism is the initial result of the work, we plan to explore more mechanisms such as the buffer management, standby peer selections and so on which constitute the possible directions of our future work.

References

1. Rejaie, R., Ortega, A.: PALS: Peer-to-Peer Adaptive Layered Streaming. In Proc. of ACM NOSSDAV, 2003.
2. Cui, Y., Nahrstedt, K.: Layered Peer-to-Peer Streaming. In Proc. of ACM NOSSDAV, 2003.
3. Malhotra, V. M., Kumar, M. P., Maheshwari, S. N.: An O(|V|3) Algorithm For Finding Maximum Flows in Networks. In Information Processing Letters, 1978.
4. Saroiu, S., Gummadi, P. K., Gribble, S. D.: A Measurement Study of Peer-to-Peer File Sharing Systems. In Porc. of MMCN, San Jose, 2002.
5. Calvert, K., Doar, M., Zegura, E.: Modeling Internet Topology. In IEEE Communication Magazine, pages 35:160-163, 1997.
6. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. 1979.