

Content Distribution for Seamless Downloading

Andreas Constantinides
Department of Electrical Engineering
Columbia University
New York, NY 10027
Email: acc40@columbia.edu
Advisor: E. G. Coffman, Jr.

I. INTRODUCTION

A new paradigm in information transmission is introduced for networks of cooperative cache servers [1]. Each file in the network has an origin server which stores the entire file, and all nodes of the network are potential clients for all files. A file segment of any size can be cached at any of the nodes. Satisfying requests is done *online* in the sense that downloading along a path from node a to node b can not begin until a has received a request to do so from b . The downloading of a file to a client is *seamless* if it incurs no delay, i.e., it begins when the request is made and continues without interruption until it is complete.

Minimal-cost seamless downloading, the new retrieval paradigm, provides the seamlessness with a minimum total amount of the file cached in the nodes of the network. This notion is based on the partial caching concept [2] which was introduced to overcome some of the disadvantages of traditional cache replacement algorithms such as LRU and LRU-threshold [3]. The main idea of partial caching is to store an initial part of the file in the cache and to obtain the rest of the file from the origin server. To realize this “perfect” retrieval performance, i.e., stream-like downloading with no initial delay, clients must be prepared to re-sequence segments of a file received out of order. Thus, if the file is a video, for example, the requesting node must experience an initial delay during which some portion of the video is buffered before streaming in the usual sense can begin.

The problem of implementing minimal-cost seamless downloading has interdependent, combinatorial *allocation* and *scheduling* subproblems. The solution to the allocation problem prescribes the file segments to be stored in different nodes in the network, and the solution to the scheduling problem specifies download schedules for each node that are based on the allocation of file segments.

II. THE SINGLE FILE PROBLEM ON TREE NETWORKS WITH SIMULTANEOUS REQUESTS

This paper is confined to the following baseline problem: *minimal-cost online seamless downloading of a single file in tree networks, with the root as the origin server, with downloading allowed only in a root-to-leaf direction, and with all file requests made at the same time. This time is taken to be $t = 0$. A node immediately forwards upstream all requests received from clients in its subtree. Downloading possibly*

different file segments from a node to more than one of its downstream neighbors can take place in parallel.

Formally, the tree network T has a root node r that stores the file F under consideration. The file $F := \{1, 2, \dots, M\}$ is a set of M distinct *units*. A unit is an arbitrary but fixed number of bits that comprises the unit of storage and transmission. Link transmission rates are all the same, and for convenience are taken to be one file unit per unit of time. Link transmission delays can vary, but are restricted to integers; they are the edge labels of the tree. Let δ_v denote the transmission delay between the non-root node v and its parent $p(v)$. If node $p(v)$ starts transmitting a unit of the file to node v at time t , then node v starts receiving it at time $t + \delta_v$. The file unit that node v can transmit downstream is either an element of its cache or has just been received from upstream.

The root r is the origin server for the file, but subsets of F are stored at the caches of different nodes in order to achieve seamlessness. Let S_v denote the set of file units cached at node v , and $U_v(t)$ denote the unit that v received from upstream at time t . “At time t ” here means that the reception is complete at time t , and took place in the interval $(t - 1, t]$. A *reception* schedule for a node v is obtained by specifying $U_v(t)$ for all t , $1 \leq t \leq M$.

The downloading is seamless if for every client node w of the tree T

$$S_w \cup \left(\bigcup_{2\delta_w + 1 \leq t \leq M} \{U_w(t)\} \right) = F \quad (1)$$

In general, apart from request signalling, interior nodes of a tree can perform two functions: they can serve as client nodes as well as serving as forwarding nodes. In order to minimize the total cache for seamless downloading, when a request is made the (local) objective of every client is to obtain as much of the file as possible from the root r . Because parallel downloading is allowed at any node, the seamless downloading problem on trees decomposes into solving *independently* the problem for each subtree of the root containing clients.

The first variant we consider is the CLIENT problem. An instance of CLIENT consists of the edge-labeled tree, an integer M denoting the number of units in F , and the subset of nodes that are clients, and request the file F at time $t = 0$.

III. A SOLUTION TO THE CLIENT PROBLEM

The seamless downloading constraint requires each client node v to store enough units in its cache to cover at least the two-way communication delay with its parent $p(v)$. Thus, the following simple, easily verified necessary condition for solutions to CLIENT is obtained.

Lemma 1: A solution to an instance of CLIENT must assign to each client node v at least $2\delta_v$ file units.

It is convenient to introduce *reduced* trees. For a given instance of CLIENT, the tree is reduced if every non-root node v is such that the subtree rooted at v has at least one client; clearly, subtrees not having clients do not participate in the seamless downloading problem. Denote the reduced tree by T' . In what follows, a cyclic shifting algorithm is described that finds reception schedules and allocations simultaneously.

A SOLUTION. Begin with an arbitrary non-root node and give it an arbitrary permutation of $1, 2, \dots, M$, its *sequence*. Now pick any node not having a sequence yet, say u , but which has a parent or a child already with its sequence. If the parent w of u has a sequence, then u 's sequence is obtained by shifting w 's sequence δ_u time units cyclically to the right, and if w is a child of u , then u 's sequence is obtained by shifting cyclically w 's sequence δ_w to the left.

After we have sequences for all non-root nodes, consider first the client nodes. The cache at one of these nodes, say v , contains simply the first $2\delta_v$ units of v 's sequence.

Finally, consider a non-client node u . If $j > 0$ is the earliest time it receives a request, delete the first and last j units of u 's sequence. Then, make the cache of u contain the next $2\delta_u$ units of its sequence, (i.e., the units $j + 1$ through $j + 1 + 2\delta_u$ of u 's sequence).

The reception schedule of a node v is easily obtained by the units remaining in its indexed sequence after the above elimination procedure. $U_v(t)$ will just be the t -th unit in v 's sequence.

Theorem 3.1: The above allocation-schedule pair achieves seamless downloading to all clients, and is also of minimal cost.

The proof that the allocation-schedule pair achieves seamless downloading is obtained by first showing that the reception schedule of any node v confines to the *online* transmission constraint. Then, seamlessness follows from the observation that the reception schedule of any client node v is just a permutation of $1, 2, \dots, M$.

The allocation is obviously of minimal cost if all nodes of the reduced tree are clients, because of Lemma 1.

To prove optimality in the case that some of the nodes of the reduced tree are non-clients, consider the path from a non-client node v to the client node u whose request is earliest to arrive at v . Let δ_{uv} be the total delay from u to the parent of v . The key observation is that to achieve seamlessness, the union of the caches in the above path must have at least $2\delta_{uv}$ *distinct* file units. The reason for this is that the file units scheduled, but not cached, at client u were downloaded just in response

to u 's request; since u was an earliest requesting node, it can not "intercept" and exploit downloading already in progress in response to the request of some other node. Our scheme allocates exactly $2\delta_{uv}$ file units along this path, and hence it is optimal.

IV. OTHER PROBLEM VARIATIONS OF INTEREST

Another problem we have considered is the ANY-CLIENT problem, an instance of which consists only of the labeled tree and M . A solution to this problem specifies a minimal total allocation of units to caches such that seamless downloading can be accomplished no matter which subset of nodes is chosen for clients (again, all requests are made at $t = 0$). A solution to this problem is not generally obtainable as a solution to CLIENT with all nodes specified as clients. (The allocation in the latter solution does not generally ensure seamlessness when any given (proper) subset of non-root nodes is chosen as the set of clients.)

Other problems we are considering include CLIENT-RELEASE in which the times at which the client nodes make their requests, (*release* times in scheduling jargon), are allowed to be arbitrary integers which become part of the problem instance.

For the CLIENT problem, as we have seen, there is an allocation which achieves seamlessness in which every non-root node v stores $2\delta_v$ units (this is the minimal cost allocation with respect to the overall cache). An interesting question we are pursuing is the following: From the set of minimal cost allocations that achieve seamlessness, which allocation also minimizes the maximum delay until streaming can begin at each client?

V. CONCLUDING REMARKS

Multiple-file problems, trees where file units can also go upstream, structures more general than trees, and non-homogeneous link transmission rates are among the many directions for future research that we are pursuing.

REFERENCES

- [1] E. G. Coffman, Jr., A. Constantinides, D. Rubenstein, B. Shepherd, and A. Stavrou, "Content distribution for seamless transmission," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 2, pp. 31–32, 2004.
- [2] S. Sen, J. Rexford, and D. F. Towsley, "Proxy prefix caching for multimedia streams," in *Proceedings of IEEE INFOCOM*, New York, March 1999, pp. 1310–1319.
- [3] M. Abrams, C. R. Standridge, G. Abdulla, S. Williams, and E. A. Fox, "Caching proxies: Limitations and potentials," in *Proceedings of 4th International World Wide Web Conference*, Boston, USA, December 1995, pp. 119–133.