# TREE DELTA TRANSCODING FOR EFFICIENT DYNAMIC RESOURCE DELIVERY ON ASYMMETRIC INTERNET

Javed I. Khan

Internetworking and Media Communications Research Laboratories
Department of Math & Computer Science
Kent State University
233 MSB, Kent, OH 44242

## Abstract

The paper presents a scheme for serving dynamic resources over the Internet. The method uses a combination of structured fragmentation of dynamic resources in a tree hierarchy and update messaging in the form of difference between the trees called the *Tree Deltas*. The proposed method reduces the impact of dynamic documents on network bandwidth, and shortens the response time. The system has been designed for an Active network like concept architecture where *Tree Delta Transcoders* (TDT) can be deployed and activated dynamically at the (active) network junction points. The system will be especially useful near the most difficult bandwidth constrained periphery of the Internet. As we shall see the analysis and results show that the proposed TDT communication can be an effective tool in surmounting the growing problem of asymmetry in the Internet in general, and in particular, can directly help in splicing wireless device networks with the commodity internet.

Keywords: *fragment streaming, tree-delta, asymmetric internetworking, transcoding.*

## 1.Dynamic Resources and Asymmetric Internet

Many resources transported over Internet face frequent update. And in majority of these cases, only a small part of the resource is generally modified. In such a situation, a mechanism that sends only the modified and altered resource segments (and a small message for deleted part), instead of retransmitting the whole, can substantially improve the notorious impact of dynamic resources on network.

Unfortunately, the major transport protocol HTTP, which is currently state-less, requires the entire resource to be retransmitted. It seems that the ability of sending content component-wise can significantly increase the efficiency of serving dynamic resources in the Internet.

The Internet is increasingly becoming populated with dynamic resources that change frequently. Typical quotes, news-sites, portfolios, community pages, web mails, personalized portals all are examples of dynamic resources with various update frequencies.

In a typical dynamic document such as the stock quote, the "detail view" or "performance view" of "stock quote" or watch document updates the document at almost every minute. However, the frequency of change varies between fields. The fields such as "last Trade", "Change", "Value Change" etc. changes almost in a few

seconds, and fields such as "Days Range" roughly changes hourly, while the fields such as "Net Asset Value", "Previous Closing" changes relatively slowly about once a day. Another example will be news web-site, where multi-megabyte portals with embedded media resources changes hourly.

Typically the modifications are localized and the effected part of the resource is quite small compared to the size of the full resource. A Banner serving portal page is yet another class of dynamic document. An extensive 1997 trace of Internet traffic indicated that approximately 20-35% of the web communication could be attributed to resources updates [6]. With the more recent explosion of Internet based commerce, and emergence of complex applications, the ratio is estimated to be much higher now and is expected to grow further at an accelerated pace.

Dynamic objects are particularly draining on network resources. The effect is multiplied, because many of the traditional optimizations do not work on them. They are difficult to cache. Therefore, most communication results in traffic on the entire path tracing back to the origin server. In the receiving end, the situation is particularly worse for devices connected to the Internet through narrow links. This includes wireless devices, where high channel error rate, low bandwidth, and high latency have paralyzing effect on the transfer of dynamic resources.

## 2.Deltas, Fragments and Related Works

The use of differential coding itself is not a new concept. It has been used to remove information redundancy in various areas ranging from video compression to software version management. Surprisingly, current web protocols have yet to take direct advantage of it. Only in last few years, several pioneering efforts have been launched to investigate the potential payoff [5,8,9,10]. The most elaborated of these initiatives focused on HTTP enhancement [5,6,7]. In late l999, Mogul et. al. presented a draft proposal recommending modification of HTTP protocol for delta encoding [4,7]. Most of these pioneering works focused more on the HTTP protocol enhancement so that an HTTP response may carry some undefined form of differential message. For actual delta computation, these initiatives relied on generic delta encoding techniques. Examples are popular Unix utility "diff" and its modifications, which can handle arbitrary binary payload. Some of the methods included "tar" like compression on top. However, such algorithms do not take the additional advantage of the underlying structure that is inherent in the bulk of its payload.

The most relevant work at payload level can be attributed to Derose and Grosso, who have investigated fragment level representation for SGLM [3]. In I999 Grosso and Veillard proposed extension in XML for facilitating fragment interchange [2, 11].

In this work, we generalize the view and focus on the entirety of the message. We demonstrate how a structured resource can be decomposed into a tree and a tree-based delta can be used for efficient web resource exchange. Indeed the ability to communicate based on logical resource fragments creates the opportunity of several high level intelligent optimization.

Another distinguishing aspect of our work is that the above proposals focused on single protocol. It seems bulk of modern communication involves multi-protocols, where one is embedded in another. However, each with it's distinct underlying structure. In this research we demonstrate a mechanism which will show how the concept of delta communication can be extended seamlessly on a tree based hierarchical model of multi-protocol structured resources, across any hard boundary of a single protocol. A particular protocol plays a role in defining only a sub-tree in the hierarchy with specified depth.

Our meta-technique is likely to be general and more widely applicable, since the crucial element, the meta-definition of a particular

protocol is expressible and communicable, whether it is a transport protocol like HTTP, or a content protocols like XHTML, within a common language construct like XML. The examples include the wide range of markup languages defined under XML, XHTML, MathML, HTML, etc. In addition, HTTP messages or multimedia data forms such MPEG-2, H.263, or GIF animations represents structured contents of varying complexities.

## 3. Tree-Delta Transcoding and Active Transcoder

For seamless integration of the fragment communication, we are investigating a particularly experimental adaptive form of fragment communication system where some of the conversions can be performed dynamically inside network at designated proxy like junction points. As we shall see, this active network [1] model allows performance optimization and dynamic rate adaptation almost to the level of individual virtual links in a network that has grossly asymmetric link capacities. It allows clients with low QoS (low bandwidth/high delay) connections (such as wireless devices) to join and share dynamic resource almost at par with the regular Internet clients.

Fragment based communication can bring multifaceted benefit. It eliminates the need to transfer the entire resource when a small part of it faces change. This reduces the number of retransmission. Also, it reduces the impact of transmission error. It also enables a number of intelligent optimizations such as *fragment streaming*. In many situations the receiver may start displaying (or processing) a complex document right after receiving the first object, while the remaining objects of the document loads. Fragment grain communication provides the users (content author, the receiving user or the application) the ability to define the loading sequence for optimum viewing flexibly, resulting into a faster and perceptually responsive system. Also in many situations the base resource may be large. Here the user may require only a fraction of it based on user interest or host limitation. Fragment communication can reduce the bandwidth consumption and response lag by allowing the browser to load only the selected part, altogether eliminating the one that will never be displayed.

## 4. System Overview

Our system (Fig-1) is composed of three parts (i) TD server agent, TD client agent, and (c) the TDT itself. The Server and Client

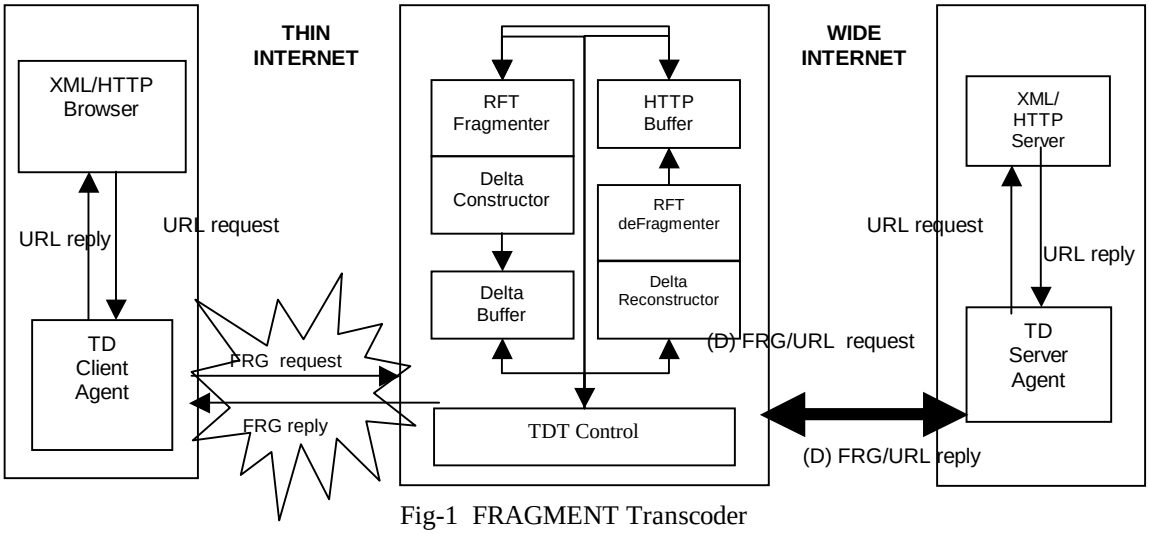


Fig-1 FRAGMENT Transcoder

agents perform optimization tasks at the end points, when such situation is available. The client agent can operate at the browser or in cache. TST sits on an active node [1].

The transcoder is designed in such a way that it can converse in both HTTP/XML as well as td-HTTP/XML (tree delta HTTP/XML) in either end. Two TDT can sit at the two ends of a thin segment of a network path to make the transformations transparent to either end-point. Fig-1 shows the placement when the TDT is placed at the edge of a commodity Internet to splice a narrow-band wireless connected client with it. Below we provide brief overview of the critical design issues:

### *4.1. Resource Fragment Tree*

The original resource is fragmented and parsed into a logical hierarchy called *Resource Fragment Tree* (RFT), $T_i$ which can have multiple protocol embedded in it. When a server agent detects change in a delta eligible document, it first generates the RFT if the modified tree $T_{i+1}$. When a resource changes, then TDT detector module identifies the location of updated fragment. Once the locations of the modified fragment(s) have been identified, the Tree-delta algorithm computes the tree difference between the original and modified RFTs. If changes take place at multiple fragments, it performs optimization to determine the best *TREE-DELTA*. Initially the TDT sends a resource sketch of the object to the client agent. During subsequent retrieval of the updated resource, the client agent includes fragment/version information from the sketch in the request. The TDT determines the fragments that need to be sent.

The Example in Fig-2 sketches the update actions on the TDTs. There can be three potential types of modification operation at any node point (i) augmentation, (ii) deletion and (iii) replacement. The messages carry the opcode, resource ID, fragment ID and optional data for later two operations.

### *4.2. State Management by Digest Banking:*

One of the principal source of inefficiency in HTTP, is the statelessness. Delta communication is inherently state-full. The server-agent can ideally handle the state information. However, we use a two step process for state maintenance that can increase communication efficiently without creating undue burden on the server.

Each resource version is represented by a *tree-digest*. Tree-digest is a compact representation of the resource, which encodes the TDT structure. It also includes information about the content (or the text represented by the nodes) however as a hash value stored in the nodes of this tree. The digest of the full TDT is used to identify the current holding of the TD-client-agent. Delta is computed from this digest. Also the response TD and is addressed with respect to the structure in the tree digest.

The latest version of the digest is always available at the client-agent and the server agent always needs to know it before the delta computation. It is best if the tree digest can be maintained in the TDT-server-agent. However, for server scalability, in this architecture we conceptually disassociate the tree digest banking from the server agent. The tree digest can also be banked at an intermediate point (such as the TD-transcoder), between the TDT client-agent and the TDT-server-agent. Consequently, in this model, the TDT-client-agent can refer to the appropriate digest simply by sending an ultra-compact resource-version tag, and TDT-server agent can retrieve the actual digest from the digest bank. A copy of the response delta is used by the digest-bank to keep itself up-to-date. Clearly, for best performance digest-bank should be co-located with the TDT-server-agent or close to it.

### *4.3. Tree Nodes:*

The logical tree node contains a hierarchical fragment ID descriptor, which uniquely identifies the fragment in the tree. For each fragment in this RFT, it uses a hash signature

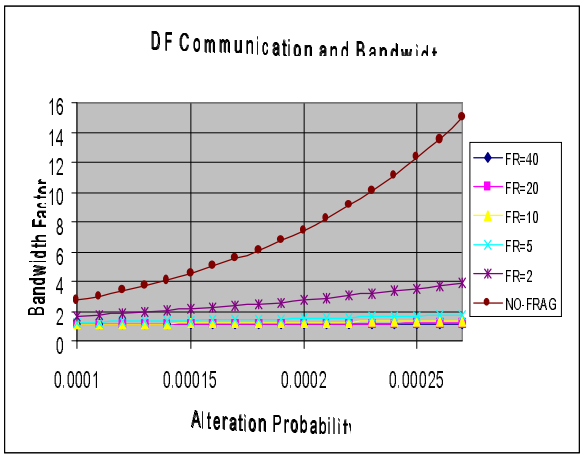


Fig-3 Bandwidth Analysis

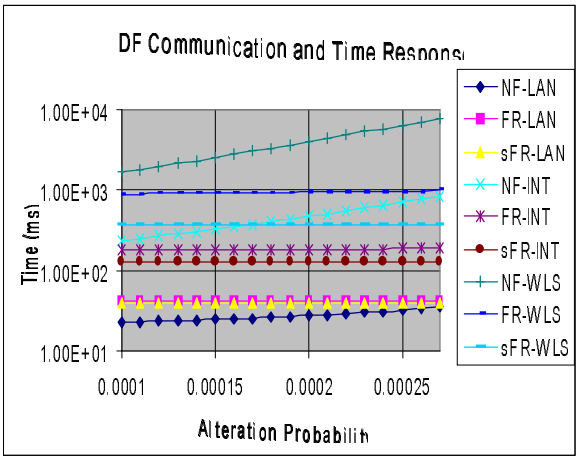


Fig-4 Time Analysis

for quick detection of changes. The actual hash function can vary. But it have to be selected to minimize the collision probability against certain types of changes such as localized addition, deletion, and spell correction types of modification.

There are also several flags in the nodes. Each node has a collision flag (CF). Forced Forward (FF) flag is used to indicate if a node requires forced re-transmission, this is useful for over active fragments that will be permanently forced served and can improve traversal algorithms. Also another flag, Skip Hash (SH) is set to indicate skip hash evaluation and force direct pattern matching. This is useful when the text content is so small that the direct evaluation is better, or for some other reason, when direct text level pattern matching have to be explicitly invoked.

## 5. Simulation Results

We have performed probabilistic modeling and simulation to identify various design parameters (such as header size, fragment grain, frequency of RFT refresh) of the transcoder systems, and to understand its overall impact on the network QoS. Below we share some preliminary results on bandwidth, response time, and fragment streaming performance.

### *5.1. Bandwidth:*

Fig-3 shows the impact of an example dynamic resource of size 10KB for various resource alteration probabilities. The x-axis shows the probability of a byte alteration. The y-axis plots the normalized bandwidth (with respect to the resource size) requirement to serve this dynamic resource.

The NO-FRG curve at the top shows the growth of bandwidth with the increased dynamism of the resource. In contrast, the curves below (FR-nn) plot the bandwidth required by Tree-Delta (TD) communication. We have plotted them for various fragment sizes expressed in the ratio of the sizes of the original resource to that of the fragment.

Visibly TD communication can dramatically improve the bandwidth requirement. As it can be seen that the explosive impact of dynamic resource update on service bandwidth can possibly be neutralized with a well designed TD-communicator.

### *5.2. Response Time:*

Fig-4 plots the response time in y-axis. We have selected three scenarios. First with 20 ms round trip time (RTT) and 10 MB/uet (per upgrade epoch transfer rate), second with 100 ms RTT and 200KB/uet per upgrade epoch transfer rate, and third with 300 ms RTT and 10 KB/uet. The values have

been selected to roughly reflect the current QoS parameters of a good LAN, Current Internet, and that of Wireless Network.

For each scenario, we have plotted performance under three communication modes. First the no-fragmentation mode indicated by "NF", then with fragmentation mode indicated by "FR". (Mode "sFR" we will explain shortly). The curves NF-INT and NF-LAN show that the response is reasonable in NF mode on a "LAN" scenario. However, the NF mode is a stressed on "INT" scenario, specially, if the resource is dynamic, and it is inadequate on WLS scenario. Note that the time scale is logarithmic. In comparison, the FR-INT and FR-WLS curves show the performance with DF-communicator in place.

The simulation added 20 ms for TD-transcoder/client-agent/ server-agent processing delay for TD measurements. As can be seen the TD scheme has two visible effects. First, it again neutralizes the effect of dynamic updates. Secondly, it dramatically rescues the service time of "WLS" type network and brings it closer to the order of "NF-INT" type network.

### *5.3. Impact on Fragment-Streaming:*

Here client can begin displaying as soon as the first fragment has been delivered. Curves denoted by "sFR" demonstrate the performance. Again, the effect is most notable for type "WLS" network where now the first displayable message comes almost as fast as it comes on a type "INT" network with no fragmentation.

### *5.4. Dynamic Optimization:*

It is also visible from Fig-4 that TD-communication may not always be profitable on type "LAN" scenarios. Indeed, the NF-LAN and FR-LAN curves reverse their order. This clearly indicates that the TD-communicator should be a dynamically or adaptively activated within the broad QoS range offered by the foreseeable communication technology suit attached to the Internet.

## 6.Conclusions

In this research, we presented a scheme for serving dynamic resources over the Internet. The method uses a combination of structured fragmentation of dynamic resources in a tree hierarchy and update messaging in the form of difference between the trees called the *Tree Deltas*. In this paper, we described the architecture of a Tree-Delta Transcoder (TDT) system, which can dynamically support the translation between the classical non-fragment non-delta (NFND) HTTP communication model and the proposed resource efficient Tree-Delta streaming model.

The proposed method reduces the impact of dynamic documents on network bandwidth. It can also shorten the response time by extending the concept of streaming to structured documents called *Fragment Streaming*. Once, individual fragments can be addressed, it becomes possible to build intelligent browsers that can optimally determine the fetch sequence of the fragments according to their visibility and rendering sequence. For large and partially occluded document, it can even eliminate fetching of many fragments.

Besides such extensions, some of the sub-techniques reported create new avenues for future investigations, as further optimization is possible.

While TDT can significantly improve application performance, their effectiveness also demonstrated the need of fundamental innovation in internetworking technology. With Active Networking, the transcoder can not only be placed at the network end-points but also can be deployed and activated dynamically on demand at the (active) network junction points. TDT can install application level intelligence and knowledge processing ability right inside network and help in creating a new generation of adaptive networks. The proposed TDT will be especially useful near the most difficult bandwidth constrained periphery of the Internet. It can potentially allow clients with low-speed connections (such as wireless

devices) to share dynamic resources almost at par with the high-speed peers.

The analysis and results show that the proposed TDT communication can be an effective tool in surmounting the growing problem of asymmetry in the Internet. While, the bandwidth numbers of the fastest portion of the networks are increasing in a routine basis due to innovations in detection and material science, however, the bandwidth numbers of the weakest portions of the Internet are relatively standing still. With the internationalization, the problem of such asymmetry is likely to intensify.

The above is part of our ongoing research where we are exploring the potential use of active trasncoding in splicing asymmetric internetwork segments as effectively as possible. The active network based model in particular allows performance optimization and dynamic rate adaptation to the level of individual links. The work is currently being funded by the DARPA Research Grant F30602-99-1-0515 under it's Active Network initiative.

## 7.References

[1] Tennenhouse., D, J. Smith, W. Sincoskie, D. Wetherall, & G. Minden, "A Survey of Active Network Research", IEEE Communications Magazine, Vol. 35, No. 1, pp 80-86. January 1997.

[2] Grosso, Paul, "XML Fragment Interchange Requirements Version 1.0", W3C Note 23-Nov-1998. [Available at: http://www.w3.org/TR/1998/NOTE-XML-FRAG-REQ-19981123].

[3] DeRose, S, Paul Grosso, "OASIS (formerly SGML Open) Fragment Interchange -- SGML Open Technical Resolution, 9601", 1996 OASIS (SGML Open) Technical Resolution, [Available at: http://www.oasis-open.org/html/techpubs.htm#fragment for an online version]

[4] Mogul, Jeffrey, B. Krishnamurthy, ,Fred Douglis, A. Feldmann, ,Y. Goland, Arthur van Hoff, "Delta Encoding In HTTP", Internet Draft RFC, 19 October 1999 [Available at: http://www.ietf.cnri.reston.va.us/internet-drafts/draft-mogul-http-delta-02.txt]

[5] Gaurav Banga, Fred Douglis, and Michael Rabinovich. Optimistic Deltas for WWW Latency Reduction. Proc. 1997 USENIX Technical Conference, Anaheim, CA, January, 1997, pp. 289-303.

[6] Fred Douglis, Anja Feldmann, Balachander Krishnamurthy, and Jeffrey Mogul. Rate of Change and Other Metrics: a Live Study of the World Wide Web. Proc. Symposium on Internet Technologies and Systems, USENIX, Monterey, CA, December, 1997, pp. 147-158.

[7] Arthur van Hoff, John Giannandrea, Mark Hapner, Steve Carter, and Milo Medin. The HTTP Distribution and Replication Protocol. Technical Report NOTE-DRP, World Wide Web Consortium, August, 1997. URL http://www.w3.org/TR/NOTE-drp-19970825.html.

[8] Jeffrey C. Mogul, Fred Douglis, Anja Feldmann, and Balachander Krishnamurthy. Potential benefits of delta encoding and data compression for HTTP. Research Report 97/4, DECWRL, July, 1997. URL http://www.research.digital.com/wrl/techreports/abstracts/97.4.html.

[9] Stephen Williams, "HTTP: Delta-Encoding Notes", 17-January-1997. Http://ei.cs.vt.edu/~williams/DIFF/prelim.html.

[10] Stephen Williams, Marc Abrams, Charles R. Standridge, Ghaleb Abdulla, and Edward A. Fox . Removal Policies in Network Caches for World-Wide Web Documents. Proc. SIGCOMM '96, Stanford, CA, August, 1996, pp. 293-305

[11] Grosso, Paul, Daniel Veillard, "XML Fragment Interchange", W3C Working Draft 1999 June 30, [Available at: http://www.w3.org/1999/06/WD-xml-fragment-19990630.html]