A LOCATION AND BANDWIDTH AWARE P2P VIDEO ON DEMAND SYSTEM FOR MOBILE DEVICES

Danilo F. S. Santos and Angelo Perkusich
Embedded System and Pervasive Computing Laboratory, Department of Electrical Engineering
Federal University of Campina Grande, C.P. 10105 - 58109-970 - Campina Grande - PB - Brazil

Keywords: Multimedia, Video on Demand, P2P Network.

Abstract: Due the increasing number of new wireless technologies and the advent of new mobile devices, new services and applications are necessary to supply the needs of these devices users. Video on-Demand (VoD) services have become one of these services. In this paper we present a proposal of a P2P Mobile Video on-Demand (VoD) System focused on mobile devices where they can act as both, media servers as well as clients. Due to different possible communication interfaces that are available for mobile devices, we considered that they can be in networks with different throughput characteristics, and also, they can be distributed through the Internet. Therefore, our proposal make possible to build content distribution trees in the application layer considering two major aspects related to mobility: the available bandwidth that the mobile device can share with the network, and; the relative position of the mobile device in the network.

1 INTRODUCTION

In the last few years, the development of new wireless technologies and the advent of embedded mobile devices, such as PDAs and smart phones, are growing rapidly. These new devices have embedded capture media devices, such as video cameras and audio recorders, therefore, users can create multimedia content. Also, with the fast growing of peer-to-peer (P2P) networks in the Internet (Wai-Pun Ken Yiu and Chan, 2007), users may share their content with other users.

As a result of this emerging scenario, new services and applications are necessary to supply the needs of the users. Video on-Demand (VoD) services have become one of these services (Thouin and Coates, 2007). The main objective for VoD is to provide users with the possibility to watch videos from the beginning at any time.

In the context of mobile devices few works provide VoD services taking into account mobility requirements. Most of the work related to VoD systems (Hu, 2001) (Ma and Shin, 2002) relies on the use of multicasting streams to share the video flow with the maximum number of clients. However, due to the nature of the Internet, multicasting is not a realistic option.

Considering the issues mentioned above, in this paper we present a proposal of a P2P VoD System focused on mobile devices where they can act as both, media servers as well as clients. Thus, they can share multimedia content, in the context of this paper video, between nodes in an overlay network. Due to different possible communication interfaces that are available for mobile devices, we considered that they can be in networks with different throughput characteristics, for instance a GPRS or a WLAN network, and also, they can be distributed through the Internet. Therefore, based on the proposal introduced in this paper it is possible to build content distribution trees in the application layer considering two major aspects related to mobility: the bandwidth available that the mobile device can share with the network, and; the relative position of the mobile device in the network.

The rest of the paper is organized as follow. In Section 2 we present a general overview of our proposal, and describe the proposed tree algorithms. In Section 3 we present a first analysis of the building tree algorithms. Finally, the conclusion is presented in Section 4.

2 PROPOSAL ARCHITECTURE

In this section we present a general overview of the proposed architecture, focusing in the media distribution protocol and its main characteristics. In the description that follows we assume that a video *V* has a duration *D* and is divided in *T* segments of *D/T* duration.

2.1 General Overview

The proposed P2P VoD system is divided in two overlay networks:

- The control network, which is responsible for the communication between client nodes and the server node that stores the original content. For the architecture a P2PSIP based network is proposed (iet, 2007).
- The distribution network, which is responsible to distribute the content (video segments) between the nodes using point-to-point connections in a P2P based way. The content is distributed using a single source application layer tree, then using a different overlay network than the previous one.

The option to use two different overlay networks was done to make possible use other mechanisms (beyond SIP) to control the distribution network. This approach is shown in Figure 1.

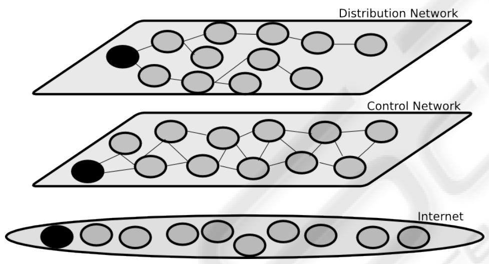

Figure 1: Overall Architecture.

2.2 Obtaining Network Information

In the proposal presented in this paper it is necessary to collect network context information to distribute the media content between mobile nodes. Specifically, our building algorithm relies on two types of information: the Available Bandwidth and the Position information. To collect this information, we propose the use of two widely-used and light-weight tools:

- Vivaldi algorithm (Dabek et al., 2004) to calculate the relative coordinates of the host in the network;
- Pathload (Jain and Dovrolis, 2002) tool to estimate the available bandwidth of a path.

2.3 Forming the Distribution Tree

After receive all requests of one layer, the server node arranges the incoming nodes in a single source tree application layer. We propose three tree building algorithms:

- *A simple bandwidth aware algorithm*;
- *A bandwidth and position aware algorithm*;
- *A fast bandwidth and position aware algorithm.*

2.3.1 Simple Bandwidth Aware Tree - SBA

This algorithm is based on the Breadth-First Search algorithm (Knuth, 1997). It uses two data structures: one which holds nodes not visited by the algorithm (*whiteList*), and other that holds visited nodes but without adjacent (children) nodes (*grayList*). The nodes that have already been visited and have children nodes are called *black* nodes.

The algorithm starts inserting the source node in the *grayList*, initializing all other nodes with it distance from the source *dsrc* as infinity and adding those in the *whiteList*. It proceeds by getting one node *X* from the *grayList*. Node *X* share bandwidth to support other *N* nodes as it children. These *N* nodes are inserted in the *grayList* and removed from the *whiteList*. Node *X* is set as a *black* node. The operation is repeated until the *grayList* is empty, so until all nodes are visited and have assigned nodes as children, if possible. This algorithm is showed in the following pseudo code.

Listing 1: SBA Pseudo Code.

```
SBA(AllNodes, srcNode)
  for each node n in AllNodes:
    color[n] := WHITE
    dsrc[n] := infinity
  end for
  color[srcNode] := GRAY
  dsrc[srcNode] := 0
  push(grayList, srcNode)
  while (grayList != 0)
    u := pullFirst(grayList)
    for i = 1 to N[u]:
      n := pullFirst(whiteList)
      color[n] := GRAY
      parent[n] := u
      dsrc[n] := dsrc[u] + 1
      push(grayList, n)
    end for
    color[u] := BLACK
  end while
```

2.3.2 Bandwidth and Position Aware Tree - BPA

This algorithm is based on the Dijkstra's algorithm (Dijkstra, 1959). It uses three data structures, the same two structures described in the previous algorithm (*whiteList and grayList*), and also a list that hold *gray* nodes with available bandwidth to support other children (*almostBlackList*).

The algorithm starts inserting the source node in the *grayList*, initializing all other nodes with it distance from the source *dsrc* as infinity and adding those in the *whiteList*. It proceeds by getting (and removing) the closest node *X* to the source from the *grayList*. This node is inserted in the *almostBlackList*. Then, for each node in the *grayList* and *whiteList* (let's merge these two lists in one *whiteAndGrayList*) it is calculated the new distance from the source element through each element in *almostBlackList*. This is the "relax" process of the Dijkstras algorithm. All visited nodes are inserted (or updated) in the *grayList*.

Listing 2: BPA Pseudo Code.

```
BPA(ALLNodes, srcNode)
  for each node n in ALLNodes:
    dsrc[n] := infinity
    color[n] := WHITE
  end for
  color[srcNode] := GRAY
  dsrc[srcNode] := 0
  insert(grayList, srcNode)
  while (grayList != 0)
    u := extract-min-dsrc(grayList)
    almostBlack := almostBlack U {u}
    p := parent[u]
    if p is not u:
      N[p] := N[p] - 1
    if N[p] = 0:
     remove[almostBlack, p]
     color[p] := BLACK
     for each node v in whiteAndGrayList:
        if parent[v] = p:
          for each node b in almostBlack:
         RELAX(b, v)
          end for
     end for
    else
      for each node v in whiteAndGrayList:
   RELAX(u, v)
      end for
  end while

RELAX(u, v)
  if (distanceOf(u,v) + dsrc[u] < dsrc[v])
    dsrc[v] := distanceOf(u,v) + dsrc[u]
    parent[v] := u
    if (color[v] = WHITE)
      color[v] := GRAY
      insert(grayList, v)
    else
      update(graylist,v)
```

When one node in the *almostBlackList* uses all *N* connections (do not support more children), then it is removed from that list and all nodes in the *whiteAndGrayList* must update its *dsrc* information (relax process). We check this every time a parent is assigned to a node. The operation is repeated until the *grayList* is empty. This algorithm is described in the following pseudo code.

2.3.3 Fast Bandwidth and Position Aware Tree - FBPA

This algorithm is based on the first algorithm presented, but using the "relax" process of the previous algorithm. Therefore, it uses two data structures, the *whiteList* and *grayList*.

As the previous algorithms, this one starts inserting the source node in the *grayList*, initializing all other nodes with it distance from the source *dsrc* as infinity and adding those in the *whiteList*. It proceeds by extracting the closest node *X* to the source node from the *grayList*. Node *X* share bandwidth to provide other *N* nodes as it children. For all nodes in the *whiteList* it is calculated the distance from *X*. Then, the *N* closest nodes from *X* are inserted in the *grayList*. Node *X* is set as a *black* node. The operation is repeated until the *grayList* is empty. This algorithm is described in the following pseudo code.

Listing 3: FBPA Pseudo Code.

```
FBPA (AllNodes, srcNode)
  for each node n in AllNodes:
    color[n] := WHITE
    dsrc[n] := infinity
  end for
  color[srcNode] := GRAY
  dsrc[srcNode] := 0
  push(grayList, srcNode)
  while (grayList != 0)
    u := pullFirst(grayList)
    for each node v in whiteList:
    dsrc[v] := distanceOf(u,v) + dsrc[u]
    end for
    for i = 1 to N[u]:
      n := extract-min-dsrc(whiteList)
      color[n] := GRAY
      parent[n] := u
      push(grayList, n)
    end for
    color[u] := BLACK
  end while
```

3 PRELIMINARY ANALYSIS

In this section we analyze the three building algorithms, comparing the mean distance from the source node against the elapsed time during the calculation of the tree. The nodes were inserted in a network topology and its coordinates were normalized to be in a Cartesian plan between (0,0) and (1000,1000). Also, the nodes could share their bandwidth with a maximun of three others nodes. The simulations were done using topologies with 10, 100, 1000, 2500, 5000, 7500 and 10000 nodes. The first graph on Figure 2 shows the mean distance from the source for each topology. The second graph on Figure 3 shows the relative elapsed time to calculate the tree for each topology. This elapsed time is relative to the calculation time of SBA, which is the fastest one.

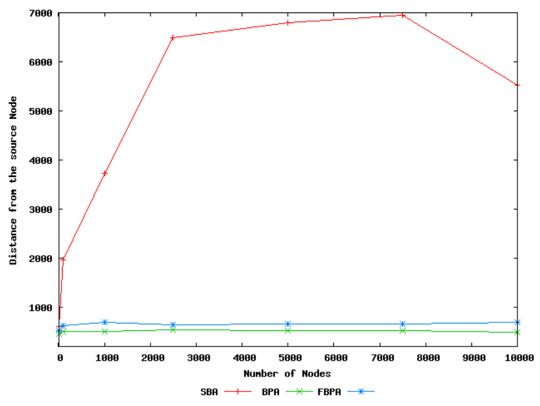

Figure 2: Mean distance from the source node.

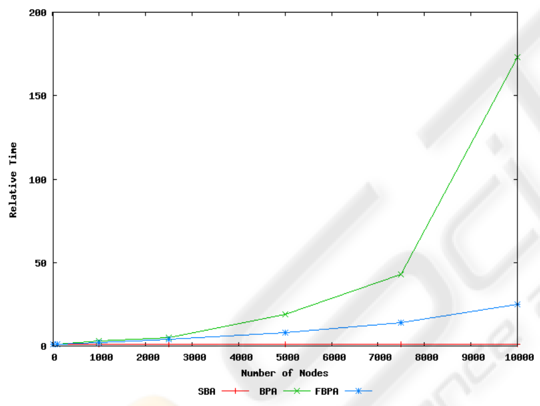

Figure 3: Relative calculation elapsed time.

Despite of the fact that the mean distance of BPA is shorter than FBPA, the relative elapsed time when calculate a tree with more than 1000 nodes of the BPA grows faster than the elapsed time of FBPA. Then, when calculating the distribution tree, a tradeoff between the number of nodes and complexity of the algorithm must be considered. Especially when dealing with a big amount of nodes (more than 1000), and nodes with restricted processing capabilities, such as mobile devices.

4 CONCLUSIONS

In this paper we presented an architecture for a mobile P2P Video on-Demand system, which uses the available bandwidth and position information of the nodes to efficiently distribute the multimedia content. It was discussed why use two overlay networks, one for the control and negation between the nodes, and another to distribute the content.

Also, we have shown three tree building algorithms that take into account both, the available bandwidth as well as the position information of the nodes to build trees with low mean distance from the source. We compared the algorithms considering the processing time to calculate the tree, and the mean distance from the source of each node. We have shown that a tradeoff between the BPA (Bandwidth and Position Aware Tree) and FBPA (Fast Bandwidth and Position Aware Tree) algorithms must be considered related to the number of nodes in each calculation.

The next step in our work is the simulation of the system as a whole in a more realistic topology. In this simulation we intend to analyze the delay for video delivery. Also, we plan to simulate others cases, such as changes of available bandwidth of a node during a transmission, leave of a node, failure of a node, change in the position of a node, and other specific cases. Finally, after the simulation stage, a full implementation of the system is planned in order to validate the system as a whole.

REFERENCES

(2007). IETF P2PSIP Working Group.

Dabek, F., Cox, R., Kaashoek, F., and Morris, R. (2004). Vivaldi: A decentralized network coordinate system.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik.

Hu, A. (2001). Video-on-demand broadcasting protocols: a comprehensive study. In *Proceedings. INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 508 – 517. IEEE.

Jain, M. and Dovrolis, C. (2002). Pathload: A measurement tool for end-to-end available bandwidth.

Knuth, D. E. (1997). *The Art Of Computer Programming*.

Ma, H. and Shin, K. G. (2002). Multicast video-on-demand services. *SIGCOMM Comput. Commun. Rev.*, 32(1):31–43.

Thouin, F. and Coates, M. (2007). Video-on-demand networks: Design approaches and future challenges. *IEEE Network*, 21:42 – 48.

Wai-Pun Ken Yiu, X. J. and Chan, S.-H. G. (2007). Challenges and approaches in large-scale peer-to-peer media streaming. In *IEEE Multimedia, Vol. 14, No. 2*, pages 50–59.