Using Symmetric Distributed Processing for Peer-to-Peer VoIP Conferencing in Auditory Virtual Environments

Philipp Berndt
sMeet Labs / Technische Universität Berlin
berndt@smeet.de

Abstract— **We present a P2P VoIP conferencing network with a symmetric distributed processing topology that we believe to be well suited for use in auditory virtual environments because it allows for multi-group communication and individual volumes per user. The network provides adaptive quality based on virtual locality while requiring significantly less bandwidth than a full-mesh topology. We also present a clustering algorithm for mapping a virtual scene to such a network structure, minimizing network diameter while respecting virtual locality. We analyzed bandwidth and latency characteristics and verified implementation feasibility by means of discrete event simulation.**

I. Introduction

With the advent of virtual environments and MMORPGs came the desire to not only hear the background sounds of the environment but also talk naturally, that is with an audio model conforming to the virtual environment. The first audio communication solutions for virtual environments were separate programs that disregarded virtual distance, orientation, room acoustics, and 3d sound altogether. Recently there have been various efforts to integrate audio communication into the virtual worlds.

A. Auditory Virtual Environments

Auditory virtual environments (*AVEs*) are a special kind of virtual environment that allows users to talk to each other in multiple groups that can be changed dynamically. Each user only hears a subset of the other users, possibly with different audio volumes.

voiscape: In [7] a multi-context voice communication system called *voiscape* is described where *"users can talk with other users and move, in a way similar to face-to-face conversation, in a virtual auditory space"*. The prototypical voiscape implementation uses peer-to-peer real-time communication. Whenever two avatars move within hearing range of each other a new bi-directional SIP/RTP connection is negotiated. This leads to a a fully-meshed communication structure as described below.

sMeet: sMeet [1] is a dynamic multi-user, multi-group telephone communication platform, where users talk to each other in a virtual environment, and the volumes with which they hear each other's voices vary subject to the current distances between their respective representations within the virtual environment.

B. Comparison of VoIP Conferencing Architectures & Related Work

The key challenge in VoIP conferencing is the distribution of audio data between participants with acceptable low latency, packet loss rates and bandwidth.

In contrast to (real-time) content distribution networks (CDNs) where content is usually streamed from one source and distributed to all consumers, VoIP conferencing deals with the streaming of several sources (speakers) from different origins to all participants.

Several audio streams can be combined into one by a process called mixing which involves digitally summing the corresponding audio samples from the incoming streams and then normalizing the result [10].

A special form of VoIP conferencing is required for AVEs where all participants are allowed to speak at once, but each one only hears a subset of the others, possibly with different audio volumes. This can only be achieved with separate mixing operations where each audio source is scaled by a factor that may be different for each listener. In some cases, however, an approximation of the scaling factors is sufficient, so that at least some mixing results can be shared by several listeners.

Various topologies can be used to distribute the audio. Table I compares the resource demands of a selection (Fig. 1), described in the following.

TABLE I

TOPOLOGY COMPARISON

resource usage \ topology	Central Server	Decoupled	Coupled	Full Mesh	Multicast
server bandwidth	high	-	-	-	-
client/peer bandwidth upstream	low	medium	medium	high	low
client/peer bandwidth downstream	low	medium	medium	high	high
server encoding effort	high	-	-	-	-
client/peer encoding effort	low	low	medium	low	low
server mixing effort	high	-	-	-	-
client/peer mixing effort	none	low	medium	high	high
individual channel control	yes	no	somewhat	yes	yes
latency	low	medium	variable	low	low
special nodes	server	root-mixer	none	none	none

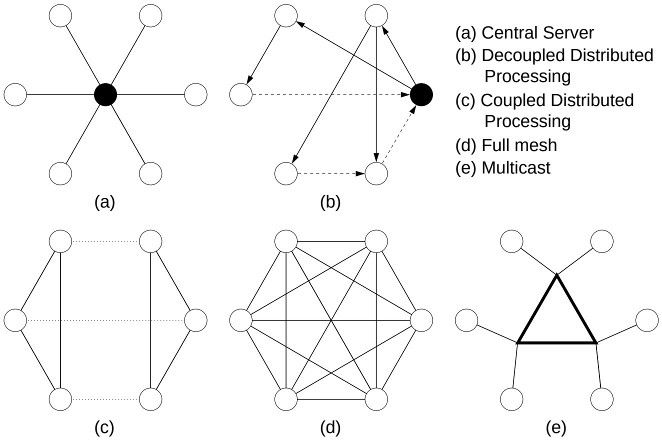
(a) Central Server
(b) Decoupled Distributed Processing
(c) Coupled Distributed Processing
(d) Full mesh
(e) Multicast
(a)
(b)
(c)
(d)
(e)

Fig. 1. Selection of distribution topologies

Central Server : The server-centric approach allows for full channel control and low latency with minimal client bandwidth but concentrates all traffic and processing cost in one point, making scalability very expensive.

Full Mesh: The most flexible topology also has the highest bandwidth requirements. As described in [8] such structure works well for small-to-medium size conferences but is less practical for bandwidth-limited end systems such as users with asymmetric DSL connections with low upstream bandwidth and does not scale well to larger conferences.

Multicast: This topology saves peer upstream bandwidth by replicating streams over a multicast backbone (in Fig. 1 denoted by the thick triangle). Unfortunately, as of today there are no multicast backbones available to the public.

Instead of distributing every source by itself, the audio streams can be mixed on the way to reduce bandwidth. The mixing is done in several stages that are performed on different nodes. This is called “distributed processing”. With P2P stream mixing this processing is done by the client nodes themselves so that no servers are required.

Decoupled distributed processing: In [3] a resource-efficient two-phased structure is presented, where the audio stream processing is decoupled into an aggregation phase that mixes audio stream of all active speakers into a single stream via a mixing tree and a distribution phase that distributes the mixed audio stream to all listeners via a distribution tree. While this allows to optimize and adapt the P2P stream mixing and distribution processes separately, it has some properties that make it less suited for AVEs:

- The voices of all participants are concentrated on one node called the root mixer. Thus it is not possible to mix for each participant all sources with different volumes.
- The delay with which each participant receives the other participants’ streams is determined by its position in the distribution tree.
- If the root mixer leaves the conference unexpectedly, a different node must take over its special responsibilities.

C. Aims and Requirements

As discussed above, providing the same high quality and low latency to all peers is costly in terms of bandwidth while using a decoupled setup leads to a situation where some nodes are treated preferentially because they occupy a position close to the root mixer in the distribution tree.

A VoIP conferencing network suitable for use in AVEs must be able to handle large groups of clients with low bandwidth, high churn, providing low latency communication and individual volume control with little or no server resources.

The limited bandwidth of each node poses a limit on the node degree while the low latency requirement demands a reasonable small network diameter.

Also, the network should provide *adaptive quality* depending on *virtual locality*: For users standing virtually close together, low latency and high audio quality and scene accuracy should be aspired. As every participant may stand virtually close to some other participant, this can only be achieved with a symmetric structure with no "per se" preferential nodes.

II. Symmetric Distributed Processing

In the this section we present a symmetric VoIP conferencing network that allows for multi-group communication and to some degree individual volumes per user. The network provides several quality levels while requiring significantly less bandwidth than a full-mesh topology, leads to an even distribution of load and has no vulnerable special nodes. One topology that exhibits these properties is the *hypercube P2P topology* presented in [11]. The proposed network uses the same topology albeit with a different communication scheme.

A. Topology

The structure is defined as follows:

Let $b \in \mathbb{N}$ denote the *base* of the topology. The base indicates how many peers form a group (i.e., are *neighbors*) on each *level* (dimension).

Let N denote the total number of peers and, as a first case, be a power of b.

Let $L = log_b N$ denote the number of levels.

Each node is then connected to $(b-1) \cdot L$ peers, i.e. to $b-1$ neighbors on each of L total levels.

Two peers, numbered i and j, $i, j \in [0, N)$, are neighbors on level ℓ iff

$$|j - i| \bmod b = 0 \ \wedge\ \lfloor \frac{i}{b^{\ell+1}} \rfloor = \lfloor \frac{j}{b^{\ell+1}} \rfloor$$

B. Communication

As illustrated in figure 2, each peer operates as follows:

```
*Record audio and send it to neighbor(s) on level 0*
*For each level $\ell$ in* $[0, L-1)$
    *Receive audio from neighbor(s) on level $\ell$,*
    *mix it with own recorded audio and audio received on levels $< \ell$ and send it to neighbor(s)*
    *on level $\ell + 1$*
*end*
*Mix audio received on levels* $[0, L)$ *and output to speaker*
```

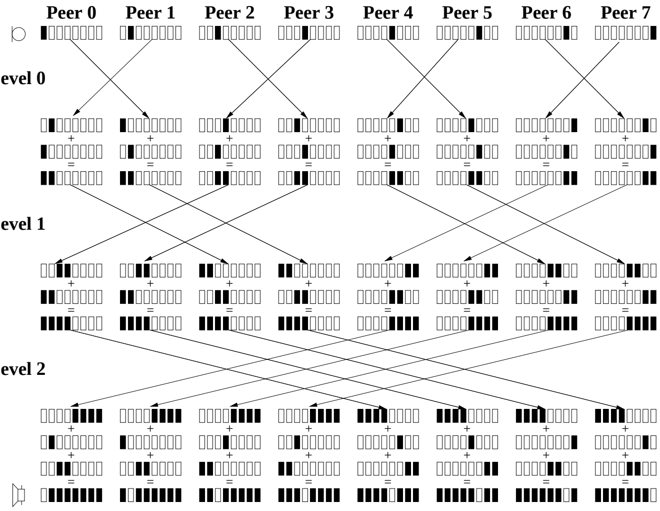

Fig. 2. Distribution structure for $b = 2$, $L = 3$. Each group of eight rectangles represents an audio packet, whereby filled rectangles represent sources that are present.

C. Properties

The latency between two peers is determined by the number of node or overlay hops, in the following referred to as *"hops"*, a packet must pass between them.

Connections to neighbors have a hop-count of $H = 1$. Because a packet can travel at most one hop per level the maximum hop-count (*network diameter*) is equal to the number of levels L.

Let $c(b, L, H)$ be the number of nodes at hop distance H from a node x. Then $c(b, L, H) = (b-1)^H \begin{pmatrix} L \\ H \end{pmatrix}$ with $H \in [1, L]$ because H hop-levels are chosen from L possible and each hop is to one of $b - 1$ neighbors.

Because of the symmetry of the structure each node's in-degree equals its out-degree and is the same for all nodes. Thus the network is a d-regular graph with

$$d = c(b, L, 1) = L \cdot (b-1) = (b-1) \cdot log_b N$$

which for $b < N$ is much less than $N - 1$, the degree of a full mesh topology.

Given a certain bandwidth limit, resulting in a maximum degree, several network configurations are possible. Figure 3 shows the cumulative hop-count distribution for all possible configurations having a degree less than or equal to 5. The n-value (applicate) of each plane represents the number of peers from which each peer receives audio with the hop-count denoted in the legend or less. The degree $d = c(b, L, 1)$ corresponds to the n-value of surface "1 hop".

As expected, the graph shows that for small conferences ($N \leq 6$), by setting $L = 1$ all peers can

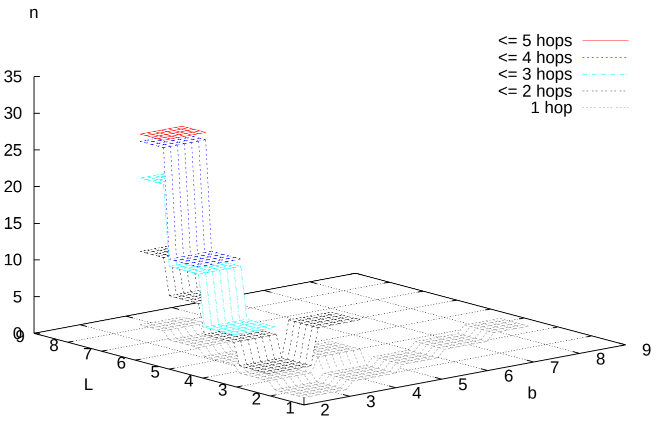

Fig. 3. Cumulative hop-count distribution for configurations with max. node degree of $d = 5$

communicate over a distance of one hop. In this case the network has the *full mesh* topology described in section I-B. For conferences with 7 to 9 participants the configuration $b = 3, L = 2$ yields the lowest hop-count. For larger conferences the most bandwidth efficient configurations with $b = 2$, $L \geq 4$ need to be used. Note that in this case each the maximum hop-count occurs for only one neighbor.

D. Handling conference sizes of $N \neq b^L$

The structures of the *incomplete* allocations $N \neq b^L$ can easily be derived by removing peers from the topologies with $N' = b^{\lceil log_b N \rceil}$. When a peer i is omitted from the network, all peers that would be sending to i on level $\ell < L$ send to former neighbors of i on level $\ell + 1$ instead. This leads to an incomplete hypercube topology.

III. Clustering

The clustering algorithm is responsible for reconciling or mapping a given scene with the virtual locations or resulting sound coefficients of the participants onto a network topology which is ignorant of geographical location, thereby assigning all participants their neighbors, i.e., clustering them into groups of size b.

A. Aims and Requirements

In order to achieve a perfectly accurate rendering of the scene according to the sound model (*perfect scene accuracy*), for each listener the audio volumes of each speaker need to be attenuated according to the listener's distance to them. Because of the proposed processing structure where mixed subsets of streams are shared, this is not possible. For its output each peer can only attenuate the composite streams it receives. The higher the level on which a composite stream is received, the more sources it contains (and are scaled by the same factor), the lower is the granularity. As stated in section II-C, the higher the hop-count is, the higher is the latency. Thus a high hop-count should correspond to a high virtual distance.

Even though perfect scene accuracy is not attainable, the algorithm should perform the clustering in an intuitive way, so that the participants' expectations (e.g., who will hear them) are approximated as closely as possible. Especially b participants standing closest to each other should

- experience a low latency between them and
- hear each other loud and clear and with a high scene accuracy.

It is desirable that participants standing further away should be heard with lower volumes. Because they blend with the background noise, however, their exact volume and latency are of lesser importance.

To keep latency and degree as low as possible, the algorithm should minimize the number of levels.

Also, if possible, the algorithm should be stable in the sense that a single person walking around in the AVE should only have local effects and should not affect the whole network structure.

Peers that are neighbors in the network experience minimum latency and transcoding artifacts between each other and peers that are neighbors on level 0 additionally have full individual channel control, i.e. highest granularity. Therefore, the algorithm proposed in the following clusters all participants standing in general position into a minimum diameter network graph with the property that participants standing closest together become neighbors on level 0.

B. Proposed Algorithm

```
$T := \{\}$
For each participant $p$
   $s :=$new singleton cluster containing $p$
   $T := T \cup \{s\}$
end
While $|T| > 1$
   $D := \{\}$
   While $|T| \geq b$
      $C :=$ smallest circle containing
         the centers of $b$ clusters
         $c_{1..b} \in T$
      $c :=$ new cluster containing $c_{1..b}$
      $D := D \cup \{c\}$
```

```
        $T := T \backslash \{c_1 \cdots c_b\}$
    *end*
    *If* $|T| \neq \{\}$
        $c :=$ *new cluster containing* $T$
        $D := D \cup \{c\}$
        $T := T \backslash \{c_1 \cdots c_b\}$
    *end*
    $T := D$
*end*
```

As distances can be determined between virtual locations as well as between gain coefficients the algorithm can be applied in either virtual or coefficient space.

C. Example

Figure 4 shows the clustering of eight participants (marked ×) to the base of three. Circle 1.1 contains the closest three points, circle 1.2 the next closest, circle 1.3 the only two points left and circle 2.1 groups the previous three clusters on the next level.

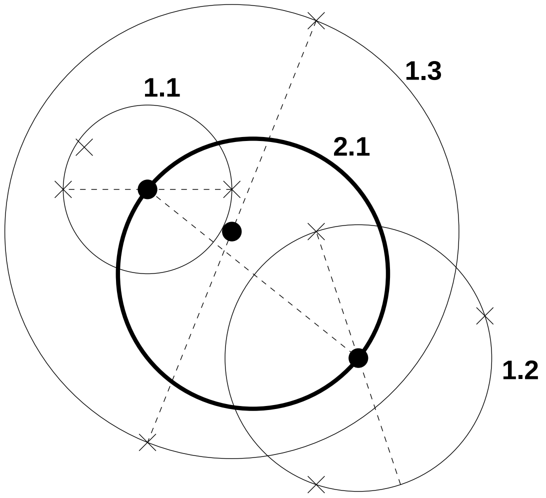

Fig. 4. Clustering 8 points (marked ×) to $b = 3$

IV. Simulation

Because for some configurations the peers' sending data rate is likely to be a limiting factor, congestion and packet queue-up may play a decisive role in the packet delays. Therefore, to survey the system dynamics, a simulation was performed.

A. Modeling Network Delay

The one-way end-to-end packet delay between two Internet users with ADSL connections can be reduced to three parts:

1) the near-end DSL up-link
2) the routing between the two DSL providers (*inter-POP delay*)
3) the far-end DSL down-link

DSL providers usually employ interleaving to increase the probability that the error correction code can compensate burst errors. For national connections this usually results in 1) and 3) being the biggest part in the end-to-end delay. For ADSL connections 1) usually incurs greater delay than 3).

Additionally, we assumed that the conferencing system will not cause noticeable Internet congestion apart from in the participants own up- and down-links. Therefore, in our simulation the influence of the conference's traffic on the Internet was disregarded.

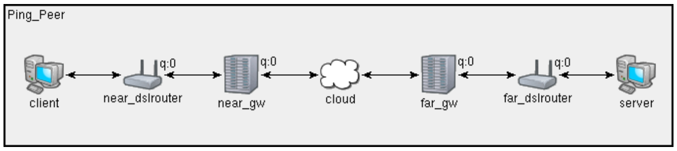

Fig. 5. Model for simulating point-to-point packet delay

Determining One-way Network Section Delay: Because of the asymmetry of the ADSL links **`bing`** [2] cannot be used for measuring both bandwidths [6]. Instead we determined the bandwidth of one direction by flooding the channel with unidirectional UDP probes of two different sizes from one side of the link and measuring the change in reception rate on the other side of the link, and then reversed sides for the other direction. The constant delay was determined by measuring the round-trip-time of a ping packet and subtracting the rate specific component.

The bandwidth measurement of the author's typical German DSL connection yielded an upstream data-rate of 208kbps and a downstream data-rate of 2,150kbps with an additional constant round-trip delay of 50ms. The one-way inter-POP delay for 1470 byte traceroute probe datagrams sent to a thousand mostly German AVE users was for at least 85% less than 10ms. These values were used as the basis for all further simulations.

B. Modeling the Peer-to-Peer Network

Each node includes the DSL delay as in the model for simulating point-to-point packet delay shown in figure 5.

Internally each peer is modeled as having one single-threaded, event-driven program ("app") that processes audio packets from the sound-card and neighbor nodes which first pass through the event-queue (and queue up if the app is busy processing) and sends out audio-packets to its sound-card and to its neighbors.

Constituting only a the small proportion of the total delay, the inter-POP delay we approximated by a simple normal distributed random variable $N(\mu, \sigma^2)$ with $\mu = 8ms$, $\sigma = 3ms$.

V. Results, Conclusions & Future Work

A prototypical JAVA implementation of the clustering algorithm for $b = 2$ worked as expected, requiring only 2ms for 128 points on an Intel Core2 CPU 6700@2.66GHz. However, the result of the clustering turned out to vary much for little changes to one point's position. With avatars moving, peers dropping out etc. the topology needs to be modified continuously. How this can be done without disrupting the user experience too much is still subject to further research.

For the simulation we assumed a VoIP codec with a bit rate of 32kbps and frame length of 10ms (such as, e.g., ITU-T G.726 [4]) and a packet header overhead of 20 bytes such as the size of a UDP header [9] plus the size of, e.g., an RTP header [12]. Figure 6 shows the mouth-to-ear delays observed by Peer0 over simulation time for configuration $b = 2$, $L = 5$ and a packet length of 50ms.[1]

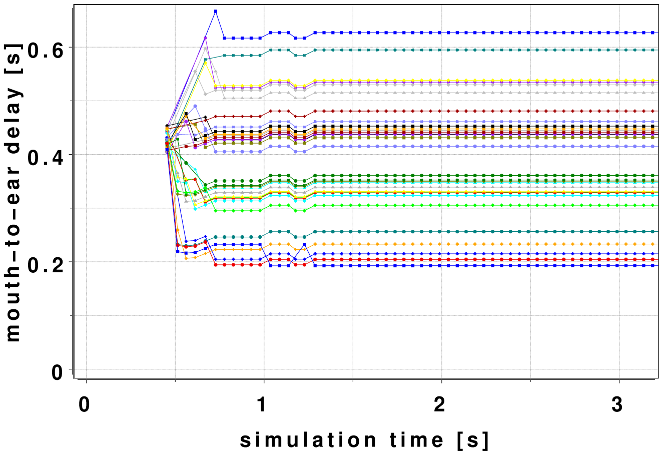

Fig. 6. Total audio delay observed by Peer0 over simulation time

As expected the delays are distributed in five bands corresponding to respective hop-counts described in section II-C. The lower two bands lie below 400ms which is considered the upper threshold of acceptable delays [5]. It can be seen that for each hop a packet travels it incurs not only network, processing and buffering delay but also delay caused by the phase displacement toward the other packets it is being mixed with. This delay is uniformly distributed with a mean of half an audio packet length. Here a synchronization of the (sub-) network could help to further reduce delay.

We have presented a P2P VoIP conferencing network and a basic matched clustering algorithm that provide the features needed for AVEs and are suited for low bandwidth conditions where full mesh communication would not be possible. The simulation results give reason to believe that the conferencing system can be implemented and will provide acceptable delay that can most likely be further reduced by future research.

Acknowledgments

The author would like to thank Dominic Battré, Peter Gober, Felix Heine and Paul McCabe for helpful comments and suggestions.

References

[1] sMeet - Reality Communications. [Online]. Available: http://www.smeet.com/

[2] P. Beyssac, "Bing - bandwidth ping," 1995.

[3] X. Gu, Z. Wen, P. S. Yu, and Z.-Y. Shae, "Supporting multi-party voice-over-ip services with peer-to-peer stream processing," in *MULTIMEDIA '05: Proc. 13th annual ACM intl. conference on Multimedia*. New York, NY, USA: ACM Press, 2005, pp. 303–306.

[4] International Telecommunication Union, "40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)," ITU, Geneva, Switzerland, Rec. G.726, Dec. 1990.

[5] International Telecommunication Union (ITU), "One-way transmission time," ITU, Geneva, Switzerland, Rec. G.114, Mar. 1993. [Online]. Available: http://www.itu.int/

[6] W. Jiang, "Detecting and measuring asymmetric links in an ip network," 1999. [Online]. Available: citeseer.ist.psu.edu/article/jiang99detecting.html

[7] Y. Kanada, "Multi-context voice communication controlled by using an auditory virtual space," Nov. 2004.

[8] J. Lennox and H. Schulzrinne, "A protocol for reliable decentralized conferencing," in *NOSSDAV '03: Proc. 13th intl. workshop on Network and operating systems support for digital audio and video*. New York, NY, USA: ACM Press, 2003, pp. 72–81.

[9] J. Postel, "User datagram protocol," Internet Engineering Task Force, RFC 768, Aug. 1980. [Online]. Available: http://www.rfc-editor.org/rfc/rfc768.txt

[10] M. Radenkovic and C. Greenhalgh, "Multi-party distributed audio service with tcp fairness," in *MULTIMEDIA '02: Proc. 10th ACM intl. conference on Multimedia*. New York, NY, USA: ACM Press, 2002, pp. 11–20.

[11] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl, "Hypercup – shaping up peer-to-peer networks," 2002. [Online]. Available: citeseer.ist.psu.edu/schlosser02hypercup.html

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for Real-Time applications," Internet Engineering Task Force, RFC 3550, July 2003. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3550.txt

[1] From an MTU of 1500 bytes follows that a packet can carry up to $l = (1,500 - 20) \cdot 8bit/32,000\frac{bit}{s} = 370ms$ of audio. With configuration $b = 2$, $L = 5$ with a node degree of 5 it follows $(300 \cdot 8bit + l \cdot 32,000\frac{bit}{s})/l \leq 208,000\frac{bit}{s}/5$, $l \bmod 10ms = 0 \Rightarrow 20ms \leq l \leq 370ms$