An Overlay Based QoS-Aware Voice-Over-IP Conferencing System[*]

Xiaohui Gu, Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign
{xgu, klara}@ cs.uiuc.edu

Rong N. Chang, Zon-Yin Shae
Network Hosted Application Services
IBM T.J. Watson Research Center
{rong, zshae}@ us.ibm.com

Abstract

While ubiquitous IP telephony has become a feasible Internet service, it is expected to meet the quality standards for traditional telephone services. This paper presents a distributed voice-over-IP (VoIP) conferencing system called Venus that is implemented as a composable application-level service overlay network. Compared to the traditional centralized approach, Venus achieves better scalability and resource utilization by efficiently aggregating resources across distributed voice mixers. Moreover, Venus provides multi-constrained quality-of-service (QoS) provisioning by establishing each conferencing session based on multiple QoS constraints (e.g., delay, loss rate) and resource requirements (e.g., bandwidth, audio channels). Venus provides failure resilient VoIP conferencing service by leveraging the fast failure recovery capability of the application-level service overlay network. Large-scale simulation results illustrate the efficiency of the Venus system.

1 Introduction

Internet has evolved into an indispensable service delivery infrastructure instead of merely providing host connectivity. IP telephony is a promising Internet service, particularly because of the significant revenue it can generate. A simple VoIP system includes two participants, where the original voice signal is periodically sampled, encoded into a bit stream, and sent over the Internet to the receiving end. In this paper, we consider an advanced VoIP service: a multi-party (or multipoint) conference service, which could include three or more participates.

The common design of a multi-party VoIP conferencing system relies on the use of a centralized multipoint control unit (MCU), which is responsible for aggregating the voices of all conference participants. However, the centralized approach suffers from the problems of: (1) scalability, where a single MCU can be short of resources (e.g., audio channels, network bandwidth) for a large conference including hundreds of participants or many concurrent conferencing sessions, (2) poor reliability due to the single point of failure, and (3) degraded quality-of-service (QoS) when most conference members are far away from the centralized MCU. Another alternative VoIP conferencing system design is to employ either IP-layer or application-layer multicast [1]. Although the multicast approach can theoretically save network bandwidth, the construction and maintenance of multiple conference trees are often too complicated for practical use, especially when we consider multi-constrained QoS requirements [1].

Hence, we propose a novel overlay based VoIP conferencing system called *Venus*. Distributed Venus nodes are interconnected into an application-level service overlay network for resource aggregation and failure resilience. Each Venus node provides both voice mixing service and application-level data routing. Given a conferencing request, the system dynamically composes a *mixing service path* consisting of a number of selected Venus nodes, based on the number and locations of conference participants, and the users' multi-constrained QoS requirements. Large-scale simulation results illustrate that Venus outperforms the centralized approach under different workload conditions. Venus also demonstrates better scaling property while we increase the number of networked voice mixers.

The rest of the paper is organized as follows. Section 2 introduces the Venus system model. Section 3 describes the mixing service path construction and maintenance algorithms. Section 4 presents the performance evaluation. Finally, the paper concludes in Section 5.

[*] This work was partially supported by the NASA grant NASA NAG 2-1406, NSF grant 9870736, 9970139. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF, NASA or U.S. Government.

[1] Previous assessment study [4] has indicated that both delay and packet loss rate greatly affect the user perceived quality of the VoIP service.

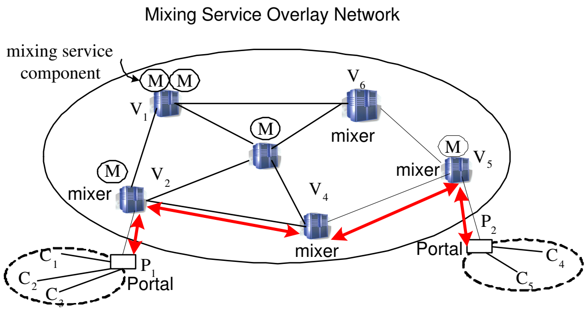

Figure 1. A mixing service overlay network.

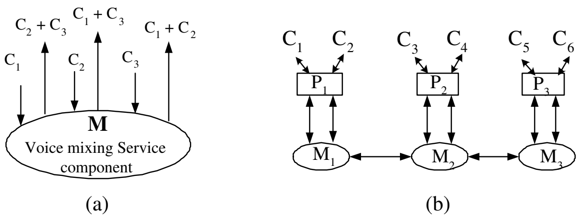

Figure 2. Mixing service component and mixing service path.

2 VoIP Conferencing System Model

We now introduce the Venus system model that consists of (1) mixing service overlay network, (2) mixing service component, (3) mixing service path, and (4) conferencing service model.

Mixing service overlay network. The mixing service overlay network (MSON) forms the Venus system's communication substrate, illustrated by Figure 1. Each overlay node called *mixer* can provide voice mixing as well as application-level data routing. For example, in Figure 1, mixers v_2 and v_5 provide voice mixing while v_4 provides application-level relaying between v_2 and v_5. A mixer can be a third-party node, which provides a service level agreement (SLA) specifying the mixing services' resource capacity (e.g., audio channels) and QoS properties (e.g., service time). For QoS management, we introduce *portals* deployed at edge networks, and *service monitors* co-located with the mixers. The portals, such as P_1 and P_2 in Figure 1, are the service access points for the conferencing clients, which collectively define the QoS assurance boundary of the Venus system. Service monitors measure the *local* resource and QoS states and report the state information to all portals. Thus, each portal can construct a global view of the MSON in terms of resource and QoS states.

Mixing service component. Each mixing service component takes k input voice signals and generates k different aggregated signals, which is illustrated by Figure 2 (a). For example, if the mixing service component has three inputs c_1, c_2, and c_3, then it generates three outputs $c_2 + c_3$, $c_1 + c_3$, and $c_1 + c_2$, which are sent back to c_1, c_2, and c_3, respectively. Each mixer can instantiate multiple mixing service components under the constraint of its resource capacity. For example, if the mixer has 20 channels and each mixing service component needs 5 channels, then it can instantiate at most 4 mixing service components.

Mixing service path. We can compose a set of mixing service components into a mixing service path[2], which is illustrated by Figure 2 (b). Although the mixing service components on the mixing service path provide the same functionality, they are different in terms of the voice content. For example, in Figure 2 (b), M_1, M_2, and M_3 have the mixed voice of c_1+c_2, c_3+c_4, and c_5+c_6, respectively. To allow each participant to hear the speeches of all other members, we must further compose the three mixing service components. We can prove that each client connected to the mixing service path receives a mixed voices of all other conference members by using the induction proof on the length of the path. Due to the space limitation, we omit the proof details here.

Conferencing service model. Each participant in a conferencing session is notified in advance with a unique conference identifier. For simplicity, we assume that all conference members of a specific conference session contact the Venus system at the same time via the portals using the conference identifier. We define the *source portal* as the portal to which the conference initiator is connected, which is responsible for composing the best mixing service path used by the conference session. The QoS metric of the conference session is defined based upon the quality measures of the mixing service path between two end portals (e.g., P_1 and P_3 in Figure 2 (b)). The rationale of the definition is that the portal-to-portal QoS is within the controllable range of the Venus system, which essentially decides the QoS perceived by conference clients. The mixing service path will be torn down at the end of the conferencing session.

3 VoIP Conferencing System Design

In this section, we present the design details of the Venus system. We first describe the QoS-aware mixing service path composition followed by the runtime mixing service path maintenance.

[2] We are aware that mixing service components can be composed into other topologies such as trees, which however requires more complicated construction and maintenance algorithms as well as extra resources for multi-level mixing. Thus, we only consider the case of composing mixing service path in this paper.

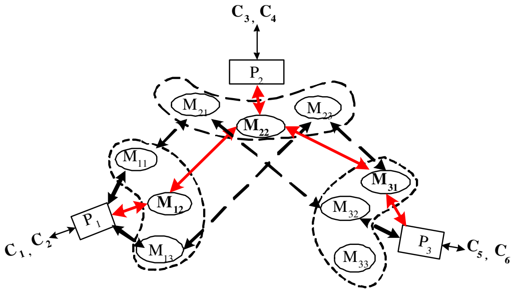

Figure 3. Illustration of mixing service path composition.

3.1 Mixing Service Path Composition

Each Venus client participates in a VoIP conference session through a Venus portal. For example, in Figure 3, clients c_1 and c_2 connect to the portal P_1; clients c_3 and c_4 connect to the portal P_2; and clients c_5 and c_6 connect to the portal P_3. Then, each portal P_i selects a number of candidate mixers for mixing voices of each client group. The candidate mixers can be selected based on a combined distance metric considering available audio channels of the mixers, network delay and loss rate from the portal to the candidate mixers. The number of candidate mixers is a configurable system parameter. For example, in Figure 3, we select three candidate mixers for each client group. Our simulation study indicates that a small number of candidate mixers (i.e., 10 mixers) can suffice the QoS provisioning goal.

Next, we need to select among candidate mixers and compose them into a mixing service path that can mix the voices of all conference participants. We formulate the problem of mixing service path composition as a multi-constrained optimal path finding problem, namely finding the best mixing service path that achieves optimal load balancing subject to the multi-constrained QoS requirements (e.g., delay and loss rate). In [2], we have proven that the above problem is NP-complete and provided a modified Dijkstra algorithm for the problem, which can achieve near-optimal performance. The key idea is to introduce an adaptive aggregated cost metric that considers multiple factors including multiple QoS metrics and load balancing objective. To satisfy multiple QoS constraints, we modify the Dijkstra algorithm by adaptively adjusting the importance weights of different factors in the aggregated cost metric based on their constraint pressures. More details of the algorithm can be found in [2].

An interesting property of composing a mixing service path is that the mixing service path can be any permutation of the selected mixing service components. For example, in Figure 3, the final mixing service path can be either $M_{12} \leftrightarrow M_{22} \leftrightarrow M_{31}$ or $M_{12} \leftrightarrow M_{31} \leftrightarrow M_{22}$. Thus, we consider all permutations of the mixing service path to further improve the QoS provisioning of the composed conferencing service. We formalize the above problem into a travelling salesman problem (TSP), which is to find a cheapest way of starting from the source portal, visiting all the selected mixers and returning to the source portal[3]. Since the TSP is also NP-complete, Venus uses a heuristic algorithm for finding the best mixing service path from all permutations.

After deciding the mixing service path, Venus instantiates a voice mixing service component for each client group on the selected mixer. Then, Venus sets up the conferencing session and notifies all conference members that the conferencing service is ready for use.

3.2 Mixing Service Path Maintenance

During runtime, the conferencing service can experience significant QoS violations or service outages due to the failures of IP-layer network links or mixers. To achieve robust VoIP conferencing service, Venus provides runtime failure detection and recovery mechanisms to maintain the availability and QoS of all active conferencing session. The source portal of a VoIP conferencing session is responsible for monitoring and maintaining the liveness and QoS of the mixing service path for each conferencing session. The failure recovery is performed at multiple layers. First, Venus relies on the overlay data routing to recover the service outage or performance failures of IP-layer network links [3]. For example, in Figure 3, if the overlay path from M_{12} to M_{22} is broken, the overlay data routing layer will dynamically find an alternative overlay path from M_{12} to M_{22}. However, overlay data re-routing cannot recover the failures of the mixers on the mixing service path (e.g., M_{12} becomes unavailable). Under that circumstance, the source portal dynamically re-composes a new mixing service path to recover the failures. To achieve fast failure recovery, we consider a localized path repair algorithm [2], which finds a new qualified mixing service path that does not include the broken mixers but has the largest overlap with the old mixing service path (i.e. the largest number of common mixers with the old mixing service path). For example, if M_{12} in Figure 3 fails, then we can recover the conferencing session by using an alternative suboptimal path $M_{11} \leftrightarrow M_{21} \leftrightarrow M_{32}$.

4 Performance Evaluation

We evaluate performance of Venus using large-scale simulations. We first use a degree-based Internet topology

[3] The cost of the edge back to the source portal is set as 0 since it is not included in the mixing service path.

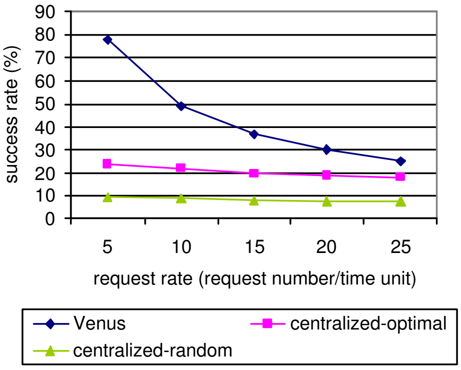

Figure 4. Session success rate comparison.

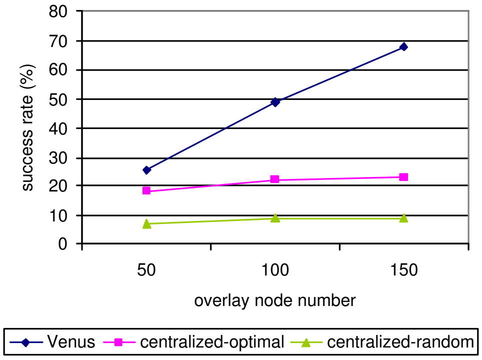

Figure 5. Scaling property comparison.

generator Inet 3.0 [5] to generate a power-law random graph topology with 3200 nodes to represent the IP-layer network. We then randomly select a number of nodes as Venus nodes and portals. The initial resource capacity and average QoS values of each network link and mixer are uniformly distributed. Each conferencing session lasts 5 to 30 time units. Each simulation runs 2000 time units. During each time unit, certain number of VoIP conferencing requests are randomly generated. Each VoIP conferencing request includes 10 to 200 participants whose locations are uniformly distributed.

We define the metric *session success rate* for performance evaluation. A QoS-aware VoIP conferencing session is provisioned successfully if and only if (1) the system has enough resources for the conferencing session, and (2) the average QoS values (i.e., delay, loss rate) measured over the whole conferencing session satisfy the required QoS values. For comparison, we also implement the *centralized-random* algorithm that randomly selects a mixer for each conference session, and *centralized-optimal* algorithm that selects the best single mixer for each conference session.

Figure 4 illustrates the average session success rate achieved by the algorithm of the Venus system presented in Section 3.1, centralized-optimal algorithm, and centralized-random algorithm, under increasing workloads on an MSON with 100 mixers. Each average success rate is measured over all conferencing requests generated during the 2000 time units simulation. We observe that Venus can achieve much higher success rates than the other two algorithms by efficiently aggregating resources of distributed mixers and finding best mixing service path under delay and loss rate constraints. Figure 5 shows the service success rate comparison under the same request rate (10 requests per time unit) on different MSONs with sizes 50, 100 and 150 nodes respectively. The results demonstrate that the Venus system presents much better scaling property than the other two approaches. The system performance increases almost linearly as we increase the number of mixers.

5 Conclusion

In this paper, we have presented a novel QoS-aware VoIP conferencing system called *Venus* using a composable application-level service overlay network. Venus achieves better QoS provisioning and resource utilization than a common VoIP conferencing system that uses a centralized MCU for a multi-party conference session. Venus can be easily deployed, which does not require any IP-layer or application-layer multicast support. Large-scale simulation results demonstrate the efficiency of the Venus system.

References

[1] J.-C. Chang and W. Liao. Application-Layer Conference Trees for Multimedia Multipoint Conferences Using Megaco/H.248. *IEEE International Conference on Multimedia and Expo (ICME 2001), Tokyo, Japan*, August 2001.

[2] X. Gu, K. Nahrstedt, R. N. Chang, and C. Ward. QoS-Assured Service Composition in Managed Service Overlay Networks. *Proc. of IEEE 23nd International Conference on Distributed Computing Systems (ICDCS 2003), Providence, RI*, May 2003.

[3] N. Kamat, J. Wang, and J. Liu. A Delay-Efficient Re-Routing Scheme for VOIP Traffic. *IEEE International Conference on Multimedia and Expo (ICME 2003), Baltimore, MD*, 2003.

[4] A. Markopoulou, F. Tobagi, and M. Karam. Assessment of VoIP Quality over Internet Backbones. *IEEE Transactions on Networking*, October 2003.

[5] J. Winick and S. Jamin. Inet3.0: Internet Topology Generator. *Tech Report UM-CSE-TR-456-02 (http://irl.eecs.umich.edu/jamin/)*, 2002.