iPTT: Peer-to-Peer Push-to-Talk for VoIP

Jiun-Ren Lin* and Ai-Chun Pang+*
*Department of Computer Science and Information Engineering
+Graduate Institute of Networking and Multimedia
National Taiwan University, Taipei, Taiwan, R.O.C.
Email: jrlin@newslab.csie.ntu.edu.tw, acpang@csie.ntu.edu.tw

Abstract

This paper proposes iPTT, a peer-to-peer Push-to-Talk (PTT) service for Voice over IP (VoIP). In iPTT, a distributed and mobile-operator independent network architecture is presented to accelerate the deployment of the PTT service. Based on the proposed architecture, the message flows for call establishment/teardown are designed to show the feasibility of iPTT. Also, we propose two mechanisms for real-time talk-burst determination, flooding-based floor control mechanism (FFC) and tree-based floor control mechanism (TFC). In terms of the determination latency and the number of floor-control message exchanges, the performance of the proposed floor control mechanisms is investigated through our analytical and simulation models.

1 Introduction

With the explosive growth of the Internet subscriber population, supporting Internet telephony service, also known as *Voice over IP* (VoIP), is considered as a promising trend in the telecommunication business. In addition to globally-deployed wired Internet telephony services [1, 2], integrating VoIP into mobile/wireless systems (e.g., 3G/GPRS, IEEE 802.11 and IEEE 802.16) is extensively studied/developed, and becomes an important issue [3, 4]. Particularly, 3GPP introduced the IP Multimedia core network Subsystem (IMS) for Universal Mobile Telecommunications Systems (UMTS) to provide real-time services such as VoIP over an all-IP network architecture [5, 6, 7].

With various wireless-VoIP applications, a walkie-talkie like service, also called Push-to-Talk (PTT), is gaining significant interest in the mobile telecommunications industry [8]. PTT is a half-duplex voice service that allows user-to-user and group communications. Unlike conventional walkie-talkie systems, the PTT service is supported by ubiquitous wireless access and thus not geographically restricted. A PTT session among a group of users is easily initiated by pressing a button, and the group members take turns talking when they obtain the floor. With PTT, a group conversation is supported, and the radio resources consumed by a multi-user call session are greatly reduced by the half-duplex voice transmission.

The existing PTT-over-cellular (PoC) solutions are provided by several mobile operators/venders [9, 10]. However, the lack of specifications and standards for PoC systems leads to a difficulty in supporting inter-operator roaming and compatibility between the user equipment of different vendors. To promote interoperability between different PTT equipment and networks, the Open Mobile Alliance (OMA) is working on developing the specifications of PoC [11]. The OMA PoC specifications utilize Session Initiation Protocol (SIP) for call signaling, and adopt a centralized architecture to support voice data broadcasting and multi-user coordination. A lot of work has been done to design and evaluate an OMA-based or proprietary PoC system [12, 13, 14, 15, 16, 17]. In an OMA-based centralized PTT architecture, a core node, i.e., PoC server, is responsible for call/floor control and voice relay. Such an architecture is intuitive and easy to implement, but the following issues should be addressed.

Scalability: The capacity of a centralized PTT system is limited to the capability of a server.

Cost: Maintenance of a standalone server incurs extra costs.

Reliability: The crash of the server results in the failure of an entire system. Also, the occurrence of congestion at the server significantly degrades system performance.

Different from the OMA-based design, a hierarchical Peer-to-Peer (P2P) service model is proposed to provide a scalable, cost-effective and robust PTT service, called iPTT. Our iPTT is implemented based on standard SIP/RTP (Real-time Transport Protocol)/RTCP (Real-time Transport Control Protocol), and does not rely on any functionalities provided by the underlying mobile networks. The network architecture and message flows for call establishment/teardown are presented to show the feasibility of iPTT. Furthermore, in iPTT, whether the real-time voice communications could be achieved depends on the efficiency of the talk-burst determination (i.e., floor control). We develop two floor control mechanisms, i.e., flooding-based floor control mechanism (FFC) and tree-based floor control mechanism (TFC), for real-time talk-burst determination over a distributed PTT system. The performance of the proposed mechanisms is investigated through our analytical and simulation models. A series of experiments are conducted to show the capabilities of our FFC and TFC.

Note that in Skype [18, 19], a P2P voice conferencing service is provided for multi-user communications. With the full-duplex transmission in Skype, a voice mixer with high-performance computing is needed. The maximum number of conferencing group members is bound by the computing power of the voice mixer [1]. Also, more radio and network resources are consumed for a full-duplex voice conferencing session than for our iPTT group communications.

The rest of this paper is organized as follows: Section 2 describes the iPTT network architecture and message flows for call establishment/teardown. Section 3 presents our floor control mechanisms for iPTT and elaborates on the detailed flows based on the proposed mechanisms. Section 4 presents the analytical and simulation models, and summarizes our experimental results to demonstrate the capability of our iPTT-based floor control mechanisms. Section 5 is the conclusion.

2 Network Architecture and Message Flows for iPTT

This section elaborates on our iPTT network architecture based on the P2P service model. Also, the message flows for signaling and voice transmission are presented

[1]Typically, the number is 5 to 10 in Skype

to show the feasibility of our iPTT system.

2.1 iPTT Network Architecture

Figure 1 shows an example of our iPTT network architecture. In iPTT, a two-level hierarchical structure is adopted to avoid excessive message exchanges among the peers. In this structure, high-level super nodes perform signaling/voice relaying and handle group-member joining/leaving. Moreover, the floor during an iPTT session is determined by the super nodes that include the session members. On the other hand, there are a large number of ordinary nodes scattered over the Internet. Each ordinary node serves as a group member, and is supervised by a specific super node. Note that in iPTT, a super node could be a member of iPTT groups, and equipped with the functionalities of an iPTT caller/callee.

When an iPTT application is executed at an user equipment, the user equipment will be an ordinary node or a super node based on its capabilities such as the bandwidth of its network connection, power consumption restriction, and computing performance. The node initialization procedure is briefly described as follows. If the node is qualified to be a super node and no appropriate super node is nearby, it will become a super node and communicate with other super nodes via Distributed Hash Tables (DHTs) [20, 21]. Otherwise, the node will find a proper super node and connect to it. The communication protocols for iPTT node initialization adopt the specifications of P2P SIP defined by IETF (Internet Engineering Task Force) [22, 23].

As shown in Figure 1, we assume that there are six nodes of the same group in our iPTT network: ON_1, ON_2, ON_3, ON_4, ON_5, and ON_6. They are supervised respectively by three different super nodes: SN_A, SN_B, and SN_C. ON_1 and ON_2 are supervised by SN_A. ON_3 is supervised by SN_B. ON_4, ON_5 and ON_6 are supervised by SN_C. In this example, the ordinary node ON_1 initiates an iPTT call session. Suppose that the call has been successfully established. The signaling procedures for call establishment are described in the following subsection. The arrows in this figure represent the voice transmission path from the call originator ON_1 to all group members (i.e., ONs 2, 3, 4, 5 and 6). The voice packets generated from ON_1 are transmitted to the destinations through the super nodes SN_A, SN_B

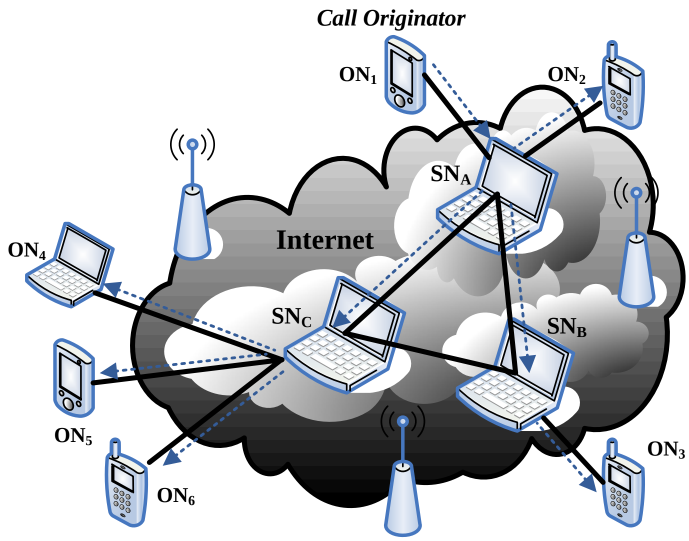

Figure 1: An Example of the iPTT Network Architecture

and SN_C. Specifically, the voice packets between SN_A and SN_C are not duplicated even though there are three group members under the supervision area of SN_C.

2.2 iPTT Signaling Message Flows

During an iPTT call session, signaling exchanges between ordinary nodes and super nodes could be divided into three stages:

Stage I (Call Establishment). ON_1 initiates an iPTT call, and issues a SIP INVITE message to the group members. At the end of this stage, voice transmission paths are established, and ON_1 broadcasts the voice message to the recipients through the paths.

Stage II (Floor Determination). Then in Stage II, ON_1 stops broadcasting, and releases the floor. The other group members that intend to talk press the button and contend for the floor via RTCP. The floor-control algorithm determines the next floor owner. One of the group members is granted the floor, and begins to speak. The floor determination process is repeatedly activated as the floor is released.

Stage III (Call Teardown). The call originator ON_1 leaves the iPTT session by sending the SIP BYE message. The call terminates, and the voice transmission path is disconnected. Note that any other members' leaving

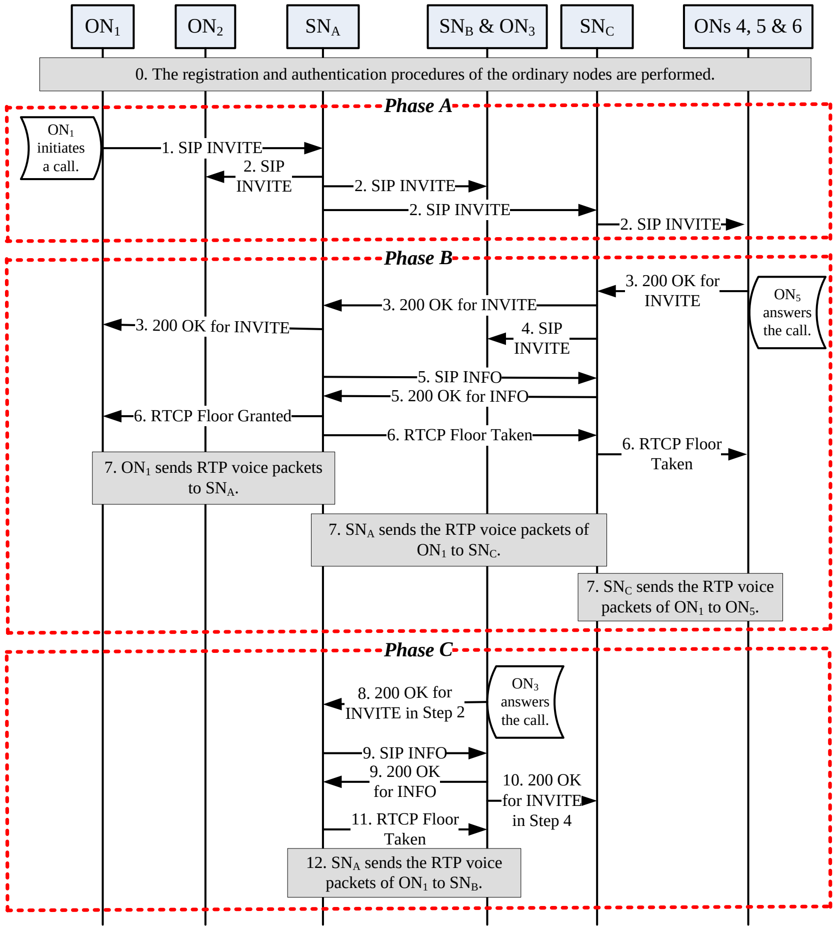

Figure 2: The Detailed Steps for Call Establishment

may lead to the voice paths' change, and will not result in the session termination.

The detailed steps for call establishment/teardown are described in the remainder of this section. The real-time floor control algorithms and the corresponding procedure are presented in Section 3.

Figure 2 shows the call-establishment procedure, where ON_1 is the iPTT call originator. Assume that all ordinary nodes registered with their super nodes and are authenticated. The message flow for call setup is described in the following steps:

Step 1: ON_1 issues a SIP INVITE message to the super node SN_A. The SIP INVITE message includes the group ID of ON_1.

Step 2: Based on the group ID, SN_A sends the SIP INVITE to those super nodes that include the group members. Then the super nodes forward the SIP

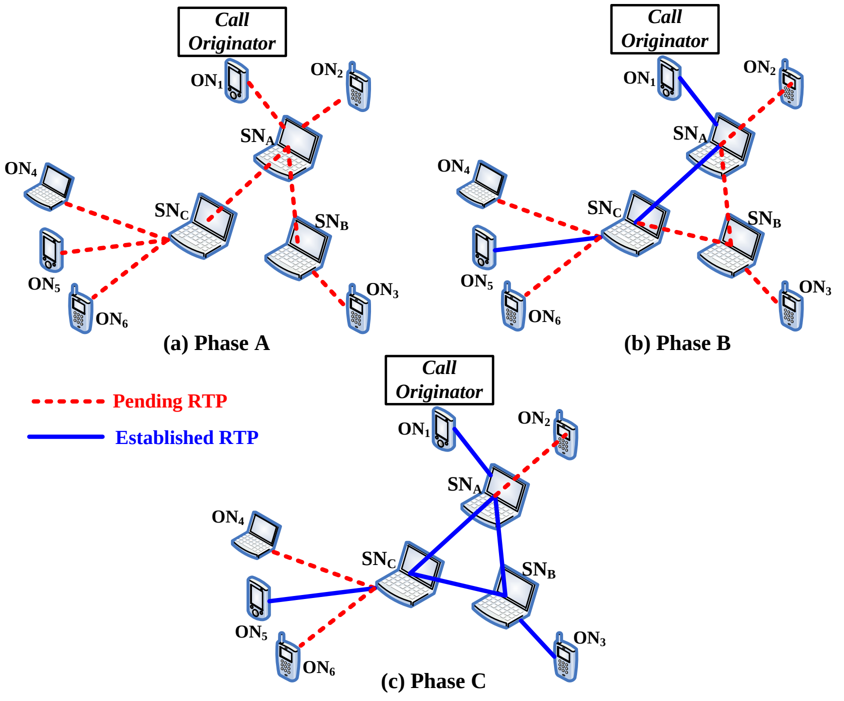

Figure 3: Three Phases for Call Establishment

INVITE to their ordinary nodes which belong to the group. Upon receipt of the SIP INVITE, the ordinary nodes ring (the 180 Ringing response is omitted in Figure 2).

In this phase, the RTP connections, i.e., the dashed lines in Figure 3 (a), are not completely established.

Step 3: Assume that ON_5 is the first node to answer the call. The 200 OK is returned to ON_1 along the path $ON_5 \rightarrow SN_C \rightarrow SN_A \rightarrow ON_1$. As shown in the solid line in Figure 3 (b), the RTP connections from ON_5 to ON_1 via SN_C and SN_A are constructed.

Step 4: A SIP INVITE message is sent from SN_C to SN_B for the upcoming RTP/RTCP connection establishment.

Step 5: If tree-based floor control mechanism (TFC) is used in iPTT, a tree structure among the super nodes covering the group members is maintained. The super node of the call originator is responsible for the tree maintenance. Upon receipt of the 200 OK response from SN_C, SN_A issues a SIP INFO message to SN_C. In this SIP INFO message, the information for the parent node of SN_C is included. In the example, the parent node of SN_C is SN_A. Otherwise, if flooding-based floor control mechanism (FFC) is used, SN_A informs SN_C of information about all participating super nodes by sending SIP INFO.

Step 6: SN_A grants the floor to ON_1 and then notifies ON_5 that the floor has been taken by ON_1.

Step 7: At this moment, ON_1 sends the voice packets to ON_5.

After Steps 3-7, the RTP connections are shown in Figure 3 (b).

Step 8: In Phase 3 (see Figure 3 (c)), ON_3 picks up the phone. An RTP path connecting ON_3 to SN_A through SN_B is established so that the voice of ON_1 could be transmitted to ON_3. If TFC is used, SN_A then adds SN_B to the tree structure , and informs SN_B of its parent node (i.e., SN_A in this example) by sending SIP INFO. Otherwise, if FFC is used, SN_A informs SN_B of information about all participating super nodes by sending SIP INFO. Also, a voice link is connected between SN_B and SN_C. Once ON_3 obtains the floor, the voice messages could be transmitted to SN_C via this voice link.

In iPTT, if the call originator leaves the session, the session is terminated. The session will go on even if other nodes leave the session. As shown in Figure 4, when ON_1 (i.e., call originator) terminates the call, ON_1 sends SIP BYE to SN_A. SN_A disconnects the RTP connection with ON_1, and responds with 200 OK. SN_A then sends SIP BYE to all participating super nodes of this session to inform them the end of this session. Upon receiving SIP BYE from SN_A, SN_B and SN_C send SIP BYE to group members in its division. Finally, the RTP paths to all group members are disconnected by using SIP BYE.

3 Real-time Floor Control

The procedure for talk-burst determination (i.e., floor control) is performed when the call originator ON_1 releases the floor. The efficiency of the floor control has a great

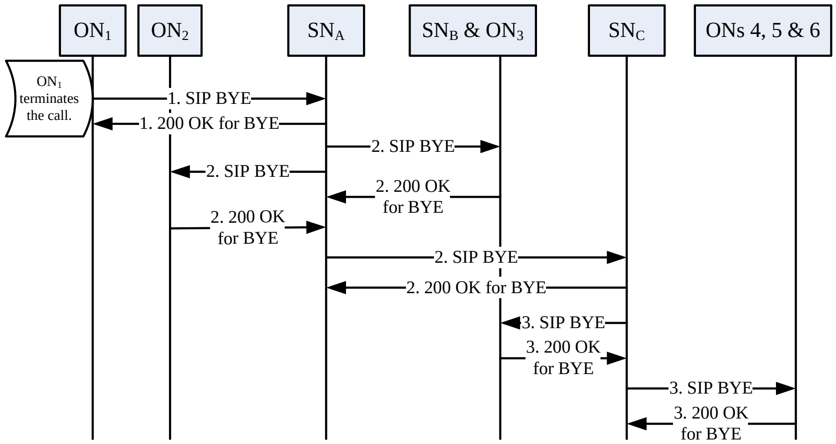

Figure 4: The Detailed Steps for Call Teardown

influence on the performance of our iPTT system. A large determination latency makes the iPTT group communications "un-smooth," and the conversations between group members are not real-time. Also, the fairness of the floor contention and the volume of signaling message exchanges of the floor-control procedure are taken into account for designing a proper determination mechanism. This section presents two talk-burst determination mechanisms: flooding-based floor control (FFC) and tree-based floor control (TFC). The determination algorithms and the corresponding message flows for these two mechanism are developed. The performance of these mechanism is investigated through our analytical and simulation models, and are described in Section 4.

Unlike an OMA-based centralized PTT network, the floor determination for a P2P PTT system is much more challenging. A considerable number of peer nodes are involved in the determination process, and an appropriate node is selected in a short period to obtain the floor. However, the complexity of the determination algorithms and the number of message exchanges are slightly reduced with our two-level hierarchical iPTT network architecture. The floor information is mostly exchanged among the super nodes equipped with higher computing/processing power. In our iPTT system, message exchanging for floor information could be done by using RTCP or SIP. Without loss of generality, we assume that RTCP messages are adopted for floor control in this paper. RTCP was originally designed for quality feedback among voice/video session users, and RTCP paths are basically the same as those for RTP. Thus it is natural to adopt RTCP for floor control to determine the next talk-burst and the voice transmission direction.

To support our iPTT floor control, RTCP messages are modified to accommodate some floor-control extensions [11]. Five RTCP messages are defined for the support of floor control: RTCP Floor Request, RTCP Floor Ack, RTCP Floor Granted, RTCP Floor Taken and RTCP Floor Release. When someone would like to contend for the floor, an RTCP Floor Request message is sent. Upon receipt of the RTCP Floor Request message, the iPTT node automatically replies with RTCP Floor Ack if the user has not requested or is not willing to request the floor. The RTCP Floor Granted and RTCP Floor Taken messages are respectively used to inform iPTT users that the floor is granted and taken by the other user. The floor owner announces the floor release to all group members through RTCP Floor Release.

In iPTT, the setting of the priorities for each floor request depends on the relative timestamp. A relative timestamp is the length of the period between the time when the RTCP Floor Release is received and the subsequent time when the floor request is made. A small relative timestamp implies that the member is more eager to get the floor and hence a higher priority is set for that request. Each time a RTCP Floor request is issued, the relative timestamp is computed and included. Besides, in order to prevent the interference from messages of different floor contention iterations, a run number is included in each floor control message. Each node in an iPTT session maintains a run counter. Whenever the floor owner releases the floor, the run counter will be increased by one. If one node gets a floor-control message that has a smaller run number than that recorded in the node, the message will be ignored. Otherwise, the message is queued and will be handled.

Before describing our flooding-based (FFC) and tree-based floor-control mechanisms (TFC), we assume that ON_2 and ON_5 both request the next talk-burst when call originator ON_1 releases the floor and sends RTCP Floor Release to all group members. Based on our iPTT hierarchical network architecture, the floor contention is divided into two levels. The local floor-control is applied to the iPTT group members residing in a single super node. The super node makes its best effort to filter unnecessary RTCP Floor Request messages of the group members and to select a candidate for the upcoming upper-level global floor control. On the other hand, the global floor-control is executed among the super nodes, where floor information can

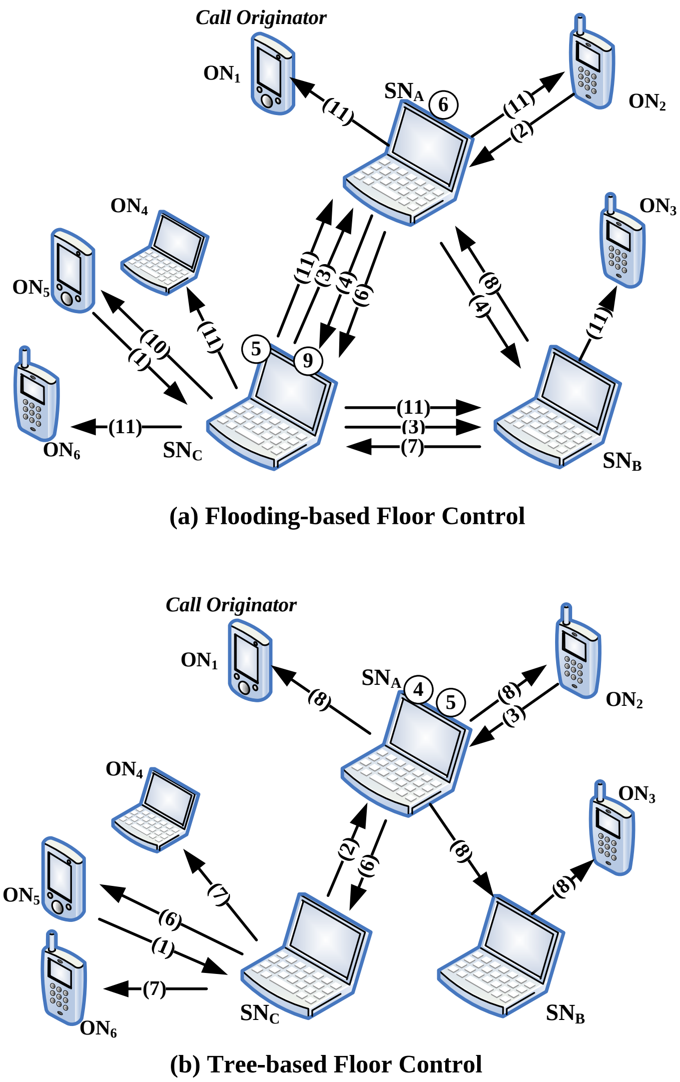

Figure 5: The Floor Control Mechanisms

be exchanged by flooding-based or tree-based transmissions. The execution steps for global floor-control mechanisms are shown in Figure 5, and described below.

3.1 Flooding-based Floor Control

In flooding-based floor control, each of the super nodes that cover floor-requesting members issues RTCP Floor Request to the remaining super nodes. Whether the receiving super nodes respond with RTCP Floor ACK depends on the intention of their group members to request the floor. If the super node that has issued the floor request receives the floor requests from other super nodes, the super node compares the relative timestamps of the requests to the one of the request it sent before. If the timestamp of the request it sent before is smaller, this request is ignored. Otherwise, RTCP Floor ACK is sent back. The super node is gained the floor only when the acknowledgements (i.e., RTCP Floor ACK) from all other super nodes are obtained. Figure 5 (a) shows the steps of FFC.

Step 1: ON_5 presses the talk button, and issues the RTCP Floor Request message (with relative timestamp T_5) to SN_C.

Step 2: ON_2 presses the talk button, and issues the RTCP Floor Request message (with relative timestamp T_2) to SN_A.

Step 3: SN_C sends RTCP Floor Request to SN_A and SN_B.

Step 4: SN_A sends RTCP Floor Request to SN_B and SN_C.

Step 5: Upon receipt of the request of SN_A, SN_C drops the request since $T_5 < T_2$.

Step 6: Upon receipt of the request of SN_C, SN_A sends RTCP Floor ACK back to SN_C.

Step 7: Since no group member of SN_B requests the floor, SN_B responds with RTCP Floor ACK after receiving the request from SN_C.

Step 8: SN_B also sends RTCP Floor ACK to SN_A to respond to the request of SN_A.

Step 9: SN_C collects the acknowledgements from all other super nodes, SN_A and SN_B.

Step 10: After collecting all acknowledgements, SN_C informs ON_5 that the floor is obtained through RTCP Floor Granted.

Step 11: SN_C notifies the super nodes and their group members (via the corresponding super nodes) that the floor is taken through RTCP Floor Taken.

3.2 Tree-based Floor Control

In the tree-based floor control, a tree structure among the super nodes is established during the call-establishment procedure (see Step 5 in Figure 2). The floor-requesting information is delivered upward to higher-level super nodes (i.e., internal nodes) in the tree structure. Each super node compares the received requests, and discards the requests with lower priorities (i.e., larger relative timestamps). Finally, the root of this tree (i.e., the super node of the call originator) determines the floor owner, and propagates this information to all super nodes following the tree structure. We notice that the tree topology may affect the performance of our tree-based floor control. For a k-ary tree ($k \geq 2$), a large k may lead to the signaling congestion of the root. Conversely, when k is small, the signaling propagation delay to the root could be large. We set $k = 2$ since the number of super nodes in our experiments is small, and the super nodes may be portable devices with limited computing hardware. As shown in Figure 5 (b), the steps of tree-based floor control are described as follows.

Step 1: This step is similar to Step 1 in Section 3.1.

Step 2: SN_C sends RTCP Floor Request to SN_A.

Step 3: This step is similar to Step 2 in Section 3.1.

Step 4: Upon receipt of the first floor request, the root (i.e., SN_A) starts to countdown a timer. The timer is set for the root to have sufficient time to collect the floor requests from super nodes.

Step 5: When the timer of SN_A is over, SN_A determines the floor owner based on the relative timestamp of the collected requests.

Step 6: Then SN_A grants the floor to ON_5 by sending the RTCP Floor Granted message via SN_C.

Step 7: SN_C also sends RTCP Floor Taken to ON_4 and ON_6.

Step 8: SN_A informs ON_1 and ON_2 that the floor is taken. ON_3 is notified by SN_A through SN_B.

4 Performance Evaluation

This section investigates the performance of our iPTT flooding-based and tree-based floor-control mechanisms. An analytical model and a discrete simulation model are developed, and a series of experiments are conducted in this section. In terms of floor-determination latency and signaling-message quantity, some numerical examples are shown to indicate the capabilities of TFC and FFC.

4.1 Input Parameters and Output Measures

In the analytical and simulation models, a network architecture with N_s super nodes and several ordinary nodes is adopted. Each super node p ($1 \leq p \leq N_s$) supervises M_p ordinary nodes ($M_p \geq 1$). Without loss of generality, $N_s = 3$ is used in the experiments[2]. $ON_{p,q}$ ($1 \leq q \leq M_p$) denotes the qth ordinary node supervised by the super node p. The propagation delay ($s_{p,i}$) of a connection between the super nodes p and i ($1 \leq p, i \leq N_s$ and $p \neq i$) follows an exponential distribution with an average $S = 200$ (ms). The propagation delay of a connection between a super node and its ordinary node can be ignored compared that that for $s_{p,i}$. The reason is that in iPTT, super nodes are widely spread out over the Internet while the ordinary nodes normally reside near their super nodes.

At each run of floor determination in iPTT, an exponentially distributed $r_{p,q}$ denotes the relative time-stamp for each floor request issued by an ordinary node $ON_{p,q}$ with an average value R. The talk-burst time for the floor owner could be a general distribution, and does not have any influence on the performance of our floor control mechanisms. When TFC is adopted for the floor determination mechanism, w_p denotes the time that the root super-node p should wait for to collect the floor requests from the other $N_s - 1$ super nodes at each floor-determination run. We assume that w_p is exponential distributed with an average $W = 200$ (ms) or 400 (ms). Note that our simulation model will be extended to accommodate Gamma distributed $s_{p,i}$ to investigate the effect of its variance.

As to the output measures, the average floor-determination latency is an important metric for our iPTT floor control mechanisms (FFC and TFC). The latency T_l is defined as the average time that a floor owner goes through to obtain the floor. In other words, T_l is the average duration between the time that the owner pushes the button and the time that he/she actually receives an RTCP Floor Granted message.

[2] In our iPTT system, the number of super nodes is much smaller than that of the members in an iPTT group.

Another important metric is the average signaling message quantity H_M during the period of a floor contention run.

4.2 Analytical Modeling

This subsection elaborates on our developed analytical model to investigate the performance of FFC and TFC. Specifically, the average waiting times (T_l) of the floor owner for our FFC and TFC are derived in the following subsections.

4.2.1 Derivation of T_l for FFC

Figure 6 shows an example of the FFC timing diagram of the floor owner for the kth determination run of an ongoing iPTT session. Assume that $ON_{p,q}$ releases his/her floor of the $(k-1)$st run at the time τ_0, and $ON_{i,m}$ obtains the floor at the coming run. The RTCP Floor Release message issued by $ON_{p,q}$ is received by $ON_{i,m}$ at the time τ_1, where $t_{p,q}$ represents the message propagation-delay from $ON_{p,q}$ to $ON_{i,m}$. $t_{p,q}$ will be 0 if $p = i$. If $p \neq i$, $t_{p,q} = s_{p,i}$. At the time τ_2, $ON_{i,m}$ pushes the button, and sends an RTCP Floor Request message with the relative timestamp $r_{i,m}$ to the super node i. Then the super node i forwards RTCP Floor Request to the other two super nodes, and receives the acknowledgements from these super nodes at the times τ_3 and τ_4. The random variables $d_{i,1}$ and $d_{i,2}$ respectively represent the delays for the RTCP Floor Request/ACK message exchange between the super node i and the other two super nodes. Upon receipt of the acknowledgements from all super nodes involving in this iPTT session, the super node i forwards RTCP Floor Granted to $ON_{i,m}$. Finally, $ON_{i,m}$ obtains the floor, and begins to talk to his/her group members. As shown in Figure 6, $ON_{i,m}$ waits for a time period to be the floor owner after pushing the talk button, and the average waiting time T_l can be expressed as

$$T_l = E[\mathsf{max}\{d_{i,1}, d_{i,2}\}]. \tag{1}$$

Based on (1), the derivations of $E[\mathsf{max}\{d_{i,1}, d_{i,2}\}]$ can be divided into two cases. In Case I (i.e., a normal case), $d_{i,1}$ and $d_{i,2}$ are respectively the round-trip delays between the super node i and the other two super nodes with the distribution functions $F_{d_{i,1}}(t)$ and $F_{d_{i,2}}(t)$, where $d_{i,1} = s_{p,i} + s_{i,p}$ and $d_{i,2} = s_{i,v} + s_{v,i}$. It is

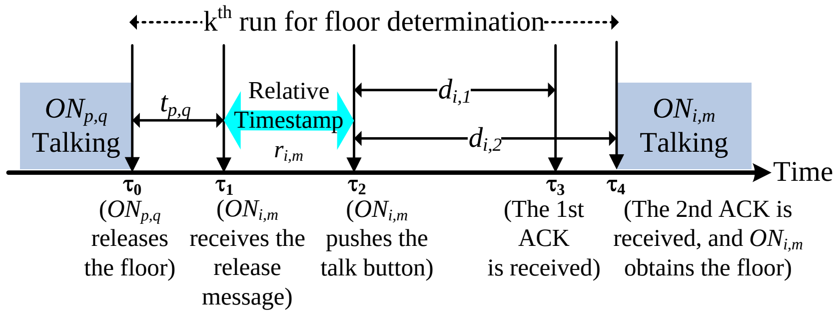

Figure 6: An Example of the Timing Diagram of FFC

obvious that both $d_{i,1}$ and $d_{i,2}$ follow an Erlang distribution with the average value $2S$. We define a random variable d_I as $\max\{d_{i,1}, d_{i,2}\}$ in this case, and $E[d_I]$ can be expressed as

$$\begin{aligned} E[d_I] &= \int_0^\infty [1 - F_{d_{i,1}}(t) F_{d_{i,2}}(t)] dt \\ &= \left(\frac{11}{4}\right) S. \end{aligned} \quad (2)$$

On the other hand, an abnormal case (Case II) occurs when any of the two super nodes (denoted as an "abnormal super node") has not received RTCP Floor Release of the previous determination run upon receipt of RTCP Floor Request of the super node i at the current floor-determination run. In this case, the "abnormal super node" has to wait for receiving RTCP Floor Release from the floor owner of the previous run. This wait operation avoids malicious iPTT users to contend the floor by advancing their request transmission before the end of the iPTT talk burst of the floor owner. To analyze $E[\max\{d_{i,1}, d_{i,2}\}]$ in Case II, the following two situations are considered.

***Case* II_a: $p \neq i$.** As shown in Figure 7 (a), the super node p issues RTCP Floor Release to the super nodes i and v. Due to the varying propagation delays of the connections, the message first arrives at the super node i. Then in Figure 7 (b), $ON_{i,m}$ requests the floor, and the super node i sends RTCP Floor Request to the super nodes p and v. The super node p responds to the super node i once it receives RTCP Floor Request. However, the super node v does not acknowledge the request of the super node i until obtaining the RTCP Floor Release message. Figure 7 (c) indicates that the

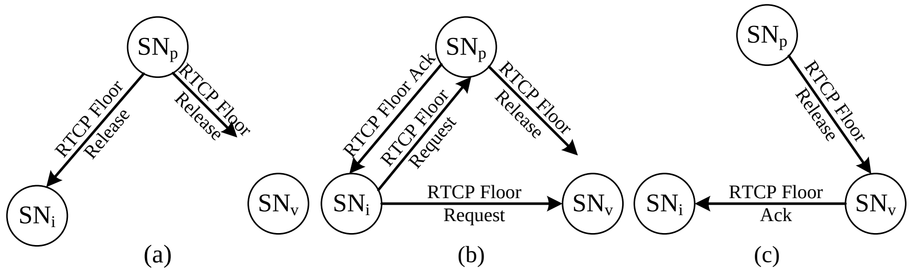

Figure 7: An Example of the Scenario in Case II_a

super node v obtains RTCP Floor Release, and sends RTCP Floor Ack back to the super node i. In this case, $d_{i,1} = s_{p,i} + s_{i,p}$, and $d_{i,2}$ is expressed as $s_{i,v} + s_{v,i} + t_{r_v}$. t_{r_v} represents the residual time for the super node v to get the RTCP Floor Release message from the super node p upon receipt of RTCP Floor Request of the super node i. With an exponentially distributed $s_{p,v}$, the residual time t_{r_v} will have the same distribution as that of $s_{p,v}$. Let d_{IIa} be a random variable $\max\{d_{i,1}, d_{i,2}\}$ in Case II_a. Then the derivation $E[d_{IIa}]$ is similar to that in (2), and

$$E[d_{IIa}] = \left(\frac{55}{16}\right) S. \tag{3}$$

Furthermore, the probability P_{IIa} that Case II_a occurs is expressed as

$$P_{IIa} \;=\; Pc_{IIa}\left\{\left(\frac{M_i}{\sum_{j=1}^{N_s} M_j}\right)\left(\sum_{p=1}^{N_s} \frac{M_p}{\sum_{j=1}^{N_s} M_j} - \frac{M_i}{\sum_{j=1}^{N_s} M_j}\right)\right\}, \tag{4}$$

where Pc_{IIa} denotes the conditional probability that an abnormal event occurs given that $p \neq i$. From Figure 7, we have

$$Pc_{IIa} = \Pr[s_{p,v} > (s_{p,i} + t_{x_i} + s_{i,v})], \tag{5}$$

where t_{x_i} represents the period from the time when the super node i forwards RTCP Floor Release to its ordinary nodes to the time when the first RTCP Floor Request is received by the super node i from one of its ordinary nodes. Then we have $t_{x_i} = min(r_{i,m}), \forall m, 1 \leq m \leq M_i$, and the

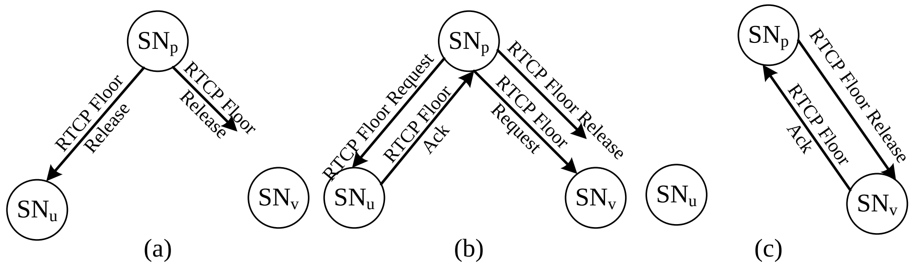

Figure 8: An Example of the Scenario in Case II_b

density function $f_{t_{x_i}}(x)$ of t_{x_i} will be

$$f_{t_{x_i}}(x) = \left(\frac{M_i}{R}\right) e^{\left(\frac{-M_i x}{R}\right)}. \tag{6}$$

Based on (6), Pc_{IIa} can be rewritten as

$$\begin{aligned} Pc_{IIa} &= \Pr[s_{p,v} > (s_{p,i} + t_{x_i} + s_{i,v})] \\ &= \frac{1}{4}\left(\frac{M_i S}{M_i S + R}\right). \end{aligned} \tag{7}$$

Case II_b***:*** $p = i$***.*** In Case II_b, the previous and current floor owners reside in the same super node. That is, the super node p issues the release messages to the super nodes u and v, and then the request messages to these super nodes after $ON_{i,m}$ pushes the talk button. If any of the super nodes v and u receives RTCP Floor Request earlier than RTCP Floor Release, it becomes an "abnormal super node". As shown in Figure 8 (b), this abnormal event occurs at the super node v. Then the abnormal super node v has to wait, and responds to the super node p immediately after receiving the RTCP Floor Release message (see Figure 8 (c)). Based on the scenario shown in Figure 8, $d_{i,1} = s_{p,u} + s_{u,p}$, and $d_{i,2}$ can be expressed as $s_{p,v} + s_{v,p} + t_{r_v}$. Then we have

$$E[d_{IIb_1}] = \left(\frac{55}{16}\right) S, \tag{8}$$

where d_{IIb_1} is the random variable of $\max\{d_{i,1}, d_{i,2}\}$ in the scenario of Figure 8. Furthermore, if the abnormal situation occurs at both the super nodes u and v, $d_{i,1}$ will be $s_{p,u} + s_{u,p} + t_{r_u}$. Then the average $E[d_{IIb_2}]$ of

d_{IIb_2} can be calculated, and will be

$$E[d_{IIb_2}] = \left(\frac{63}{16}\right) S. \tag{9}$$

The probabilities P_{IIb_1} and P_{IIb_2} that Case II_b occurs are expressed as

$$P_{IIb_1} = Pc_{IIb_1} \left\{ \left(\frac{M_i}{\sum_{j=1}^{N_s} M_j} \right)^2 \right\}, \tag{10}$$

$$P_{IIb_2} = Pc_{IIb_2} \left\{ \left(\frac{M_i}{\sum_{j=1}^{N_s} M_j} \right)^2 \right\}, \tag{11}$$

From Figure 8, we have

$$\begin{aligned} Pc_{IIb_1} &= \{\Pr[s_{p,v} > (t_{x_p} + s_{p,v})]\{1 - \Pr[s_{p,v} > (t_{x_p} + s_{p,v})]\}\} \\ &\quad + \{\Pr[s_{p,u} > (t_{x_p} + s_{p,u})]\{1 - \Pr[s_{p,u} > (t_{x_p} + s_{p,u})]\}\}, \end{aligned} \tag{12}$$

$$Pc_{IIb_2} = \Pr[s_{p,v} > (t_{x_p} + s_{p,v})] \Pr[s_{p,u} > (t_{x_p} + s_{p,u})], \tag{13}$$

Then Pc_{IIb_1} and Pc_{IIb_2} can be rewritten as

$$Pc_{IIb_1} = \left(\frac{M_i S}{M_i S + R}\right) \left[1 - \frac{1}{2}\left(\frac{M_i S}{M_i S + R}\right)\right], \tag{14}$$

$$Pc_{IIb_2} = \left[\frac{1}{2}\left(\frac{M_i S}{M_i S + R}\right)\right]^2, \tag{15}$$

Based on (2),(3),(4),(8),(9),(10),and (11) listed above, the average waiting time T_l can be derived as

$$\begin{aligned} T_l &= \left\{ \left[1 - \sum_{j=1}^{3} (P_{IIa} + P_{IIb_1} + P_{IIb_2}) \right] E[d_I] \right\} \\ &\quad + \left\{ \sum_{j=1}^{3} \{ P_{IIa} E[d_{IIa}] + P_{IIb_1} E[d_{IIb_1}] + P_{IIb_2} E[d_{IIb_2}] \} \right\} \end{aligned} \tag{16}$$

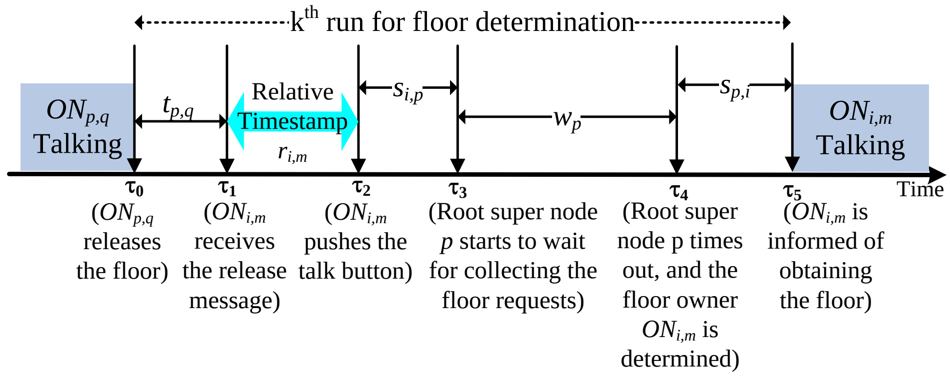

Figure 9: An Example of the Timing Diagram of TFC

4.2.2 Derivation of T_l for TFC

Figure 9 shows an example of the TFC timing diagram of the kth floor owner for an ongoing iPTT session. Assume that $ON_{p,q}$ releases his/her floor of the $k-1$st run at the time τ_0, and $ON_{i,m}$ obtains the floor at the coming run. Without loss of generality, it is assumed that the super node p plays the role of the root super node in this example. The RTCP Floor Release message issued by $ON_{p,q}$ is received by $ON_{i,m}$ at the time τ_1. Then $ON_{i,m}$ pushes the talk button, and sends RTCP Floor Request out at the time τ_2. Upon receipt of RTCP Floor Request, the root super node p starts its timer w_p to count down. When the timeout event occurs at τ_4, the super node p has collected the floor requests from the ordinary nodes, and determines that $ON_{i,m}$ obtains the floor. At the time τ_5, $ON_{i,m}$ is informed of that, and begins to talk. To analyze T_l for TFC, the following two cases are considered. If the super node of the floor owner is root, then T_l can be expressed as W. On the other hand, when $p \neq i$ (i.e., the floor of the kth run is granted to the ordinary node that is not supervised by the root super node p), then we have $T_l = 2S + W$. The probability P_a for the case of $p = i$ is derived as follows.

First, V_j is defined as the probability that the relative timestamp of the request from the root super node is smaller than the one of the request from the super node j $(2 \leq j \leq N_S)$. Then we have

$$V_j = \frac{M_p}{M_p + M_j}. \tag{17}$$

It is obvious that the root super node will get the floor when the above event occurs.

However, if the relative timestamp of the root is larger than that of the super node j, the root super node p can still get the floor when the request of j cannot arrive the root super node before timeout. We define Q_j as the probability that the j's request with a smaller relative timestamp can not arrive the root super node p before timeout. Q_j can be derived as

$$\begin{aligned} Q_j &= \Pr[w_p + t_{x_p} < s_{j,p} + t_{x_j} + s_{p,j} \text{ ,and } t_{x_p} > t_{x_j}] \\ &= \frac{M_j M_p}{M_j + M_p}\Bigg\{\left[\frac{R}{M_p}\right] + \left[\frac{S^2 R}{(S+W)(SM_p+R)}\right] \\ &\quad - \left[\frac{S^3R^2}{W(S+W)(SM_p+R)^2}\right] - \left[\frac{S^3 R}{(S+W)^2(SM_p+R)}\right]\Bigg\}. \end{aligned} \tag{18}$$

Then P_a can be

$$P_a = \prod_{j=2}^{N_s}(V_j + Q_j), \tag{19}$$

Finally, T_l for our TFC can be expressed as

$$T_l = WP_a + (2S+W)(1-P_a) \tag{20}$$

4.3 Simulation and Numerical Results

This subsection develops a simulation model to investigate the performance of our iPTT floor-control mechanisms. Our simulation program follows the discrete-event model with the input parameters and output measures presented in Section 4.1.

Figure 10 plots T_l obtained from our developed mathematical analysis and simulation experiments for FFC and TFC. From this figure, the analytical and experimental results are consistent, and our simulation has been validated against the mathematical analysis. Figure 10 shows the effect of the request arrival rate ($1/R$) on the floor determination latency T_l for our flooding-based and tree-based floor control mechanisms. In this figure, we observe that as the request-rate increases, T_l for TFC decreases and T_l for FFC increases. For TFC, a large request-rate for an ordinary node results in the increase of the probability that the ordinary nodes supervised directly by the root obtain the floor. On the other hand, more frequent

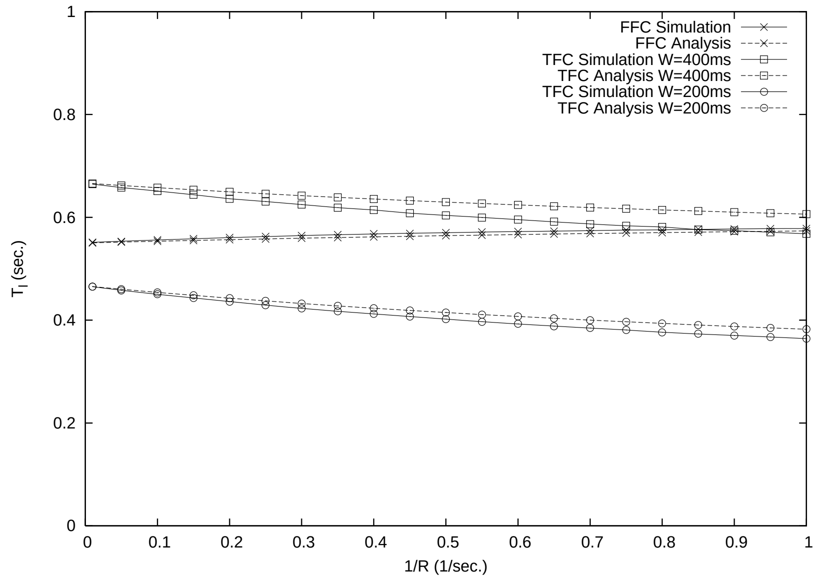

Figure 10: Effect of Request Rate on Average Waiting Time

floor-requesting in FFC leads to higher determination overhead, and thus T_l increases. When $W = 200ms$, for all arrival rates under investigation, FFC has a larger T_l than TFC. When $W = 400ms$, T_l of TFC is larger than that of FFC for $\frac{1}{R} \leq 0.8$. However, an opposite result is observed for $\frac{1}{R} \geq 0.8$. Figure 10 also indicates that the decreasing rate of T_l for TFC is larger for a small W than for a large W, which implies that the ordinary nodes directly supervised by the root benefit by a short waiting timer and an unfair situation would raise. Figure 11 shows the effect of the request rate $(1/R)$ on the number H_M of signaling message exchanges for FFC and TFC. This figure indicates that for all request rates under investigation, TFC has a smaller H_M than FFC. As the request rate increases, the total number of request messages increases, and hence H_M of TFC and H_M of FFC both increase. To further investigate the effect of variances of propagation delays between the super nodes on average waiting time T_l, a Gamma distributed random variable $s_{p,i}$ is adopted. Figure 12 indicates that the waiting time T_l of FFC increases as the variance v_s of $s_{p,i}$ increases. Specifically, the increasing rate is larger for a larger v_s than that for a small v_s. For FFC, each floor-requesting super node has to wait for acknowledgements from all other super nodes before obtaining the floor. As v_s increases, there are more probably extremely long propagation delay between the super nodes, which results in the increase of T_l in FFC. On the other hand, we

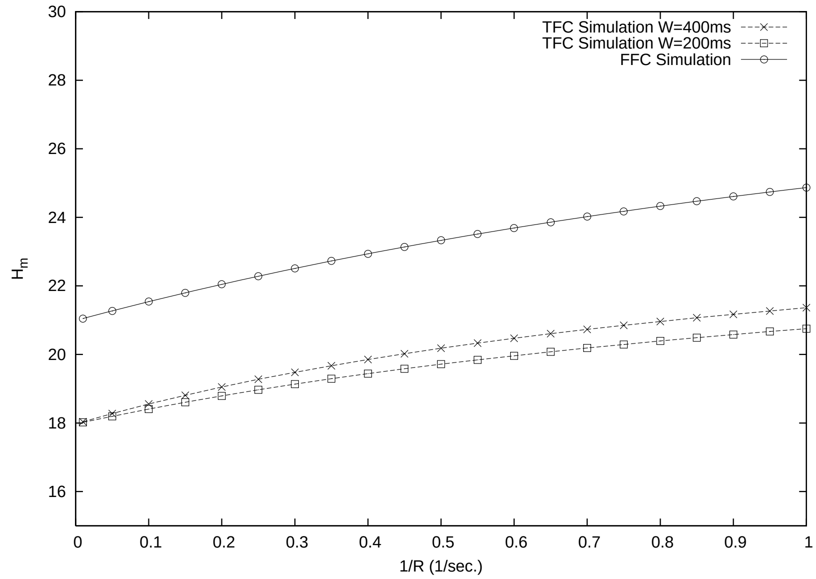

Figure 11: Effect of Request Rate on Average Number of Signaling-Message Exchanges

observe that the waiting time T_l of TFC drops slightly when the variance is large. In TFC, the timer of the root super node starts to count down at the time that the first request arrives. A large v_s implies that the first request arrives the root in a very short period, and the timer can be quickly triggered. From this figure, TFC outperforms FFC when the network is in an unstable situation with much varying propagation delays.

5 Conclusion

In this paper, we proposed iPTT, a peer-to-peer Push-to-Talk (PTT) service for Voice over IP. In iPTT, a distributed and mobile-operator independent network architecture was presented to accelerate the deployment of the PTT service. Based on the proposed two-level hierarchical architecture, the message flows for call establishment/teardown were designed to show the feasibility of iPTT. Also, we presented two floor control mechanisms, FFC and TFC, for real-time talk-burst determination. The performance of the proposed floor control mechanisms was investigated through our analytical and simulation models in terms of the determination latency and the number of floor-control message exchanges. A series of experiments are conducted to show the capabilities of our FFC and TFC.

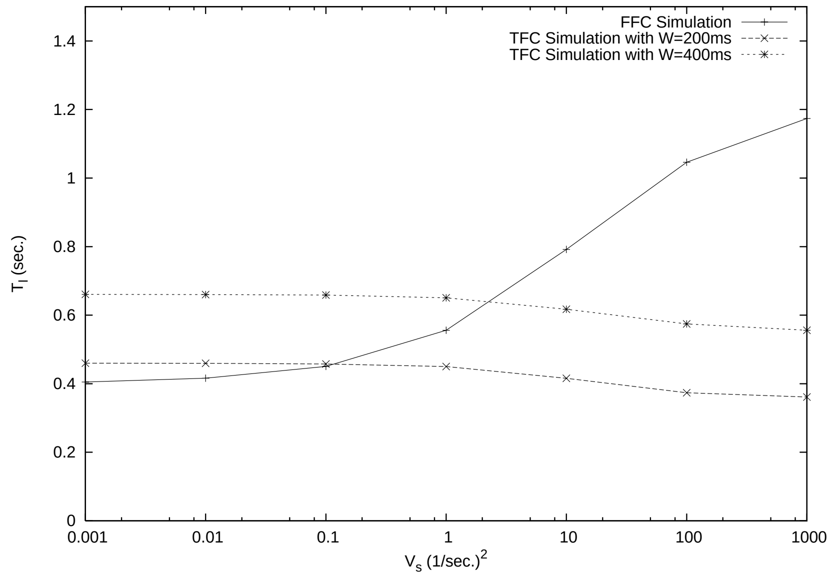

Figure 12: Effect of Variance of Propagation Delays Between Super Nodes on Average Waiting Time

References

[1] Google, "Google Talk." http://www.google.com/talk/.

[2] Microsoft, "MSN Messenger." http://messenger.msn.com.

[3] Wang, X.-G., Min, G., And Mellor, J.E., "Improving VOIP application's performance over WLAN using a new distributed fair MAC scheme," *18th International Conference on Advanced Information Networking and Applications*, vol. 1, pp. 126–131.

[4] Anjum, F., Elaoud, M., Famolari, D., Ghosh, A., Vaidyanathan, R., Dutta, A., Agrawal, P., Kodama, T., And Katsube, Y., "Voice performance in WLAN networks - an experimental study," *Global Telecommunications Conference*, vol. 6, pp. 3504–3508, Dec. 2003.

[5] 3GPP, "3rd Generation Partnership Project; Technical Specification Group Services and Systems Aspects; Service Requirement for the Internet Protocol (IP) Multimedia core network subsystem; Stage 1." Technical Specification 3G TS 22.228 version 7.3.0 (2005-12), 2005.

[6] 3GPP, "3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Signalling flows for the IP multimedia call control based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3." Technical Specification GPP TS 24.228 version 5.14.0 (2005-12), 2005.

[7] Lin, Y.-B. And Pang, A.-C., *Wireless and Mobile All-IP Networks.* John Wiley & Sons, Inc., 2005.

[8] DaSilva, L.A., Morgan, G.E., Bostian, C.W., Sweeney, D.G., Midkiff, S.F., Reed, J.H., And Thompson, C., "The Resurgence of Push-to-Talk Technologies," *Communications Magazine, IEEE*, vol. 44, pp. 48–55, Jan. 2006.

[9] Nextel, "Nextel Walkie-Talkie." http://www.nextel.com.

[10] Motorola, "Motorola Push-to-Talk over Cellular Solution." http://www.motorola.com/networkoperators/pdfs/new/PoC.pdf.

[11] OMA, "Push to Talk over Cellular (PoC) - Architecture." OMA-AD-PoC_V1_0-

20060127-C Candidate Version 1.0 V 27 Jan 2006.

[12] Parthasarathy, A., "Push to talk over Cellular (PoC) Server," *Networking, Sensing and Control, 2005. Proceedings. IEEE*, pp. 772–776, Mar. 2005.

[13] Kim, P., Balazs, A., van den Broek, E., Kieselimann, G., And Bohm, W., "IMS-based push-to-talk over GPRS UMTS," *Wireless Communications and Networking Conference*, vol. 4, pp. 2472–2477, Mar. 2005.

[14] Wu, L.-Y., Tsai, M.-H., Lin Y.-B., And Yang J.-S., "A Client-Side Design and Implementation for Push to Talk over Cellular Service," *Accepted and to appear in Wireless Communications & Mobile Computing.*

[15] Tsai, M.-H., And Lin Y.-B., "Performance Evaluation of PoC Talk Burst Control Mechanism," *Accepted and to appear in IEEE Transactions on Mobile Computing.*

[16] Balazs, A., "QoS in wireless networks: Push-to-talk Performance over GPRS," *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pp. 182–187, Oct. 2004.

[17] Raktale, S.K., "3PoC - An Architecture for Enabling Push to Talk Services in 3GPP Networks," *IEEE International Conference on Personal Wireless Communications, ICPWC*, pp. 202–206, Jan. 2005.

[18] Gao, L. And Luo, J., "Performance Analysis of a P2P-Based VoIP Software," *International Conference on Internet and Web Applications and Services/Advanced International Conference on Telecommunications*, vol. 00, pp. 11–11, Feb. 2006.

[19] Baset, S.A. And Schulzrinne, H., "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol," *Accepted and to appear in IEEE Infocom 2006.*

[20] Dury, A., "Auto-adaptive Distributed Hash Tables," *Grid Computing, 2005. The 6th IEEE/ACM International Workshop*, November 2005.

[21] Landsiedel, O.; Lehmann, K.A.; Wehrle, K., "T-DHT: Topology-Based Distributed Hash Tables," *Peer-to-Peer Computing, 2005. P2P 2005. Fifth IEEE International Conference*, pp. 143–144, September 2005.

[22] Shim, E.; Narayanan, S.; Daley, G., "An Architecture for Peer-to-Peer Session Initiation Protocol (P2P SIP)." IETF Internet Drafts, February 2006.

[23] Willis - Ed., D.; Bryan, D.; Matthews, P.; Shim, E., "Concepts and Terminology for Peer to Peer SIP." IETF Internet Drafts, June 2006.