Adaptive playout scheduling algorithm tailored for real-time packet-based voice conversations over wireless ad-hoc networks

Sofiene Jelassi*, Habib Youssef
Research Unit PRINCE, Higher Institute of Computer Science and Technologies of Communication, Hammam Sousse, Tunisia

Abstract
The effective provision of real-time, packet-based voice conversations over multi-hop wireless ad-hoc networks faces several stringent constraints not found in conventional packet-based networks. Indeed, MANETs (mobile ad-hoc networks) are characterized by mobility of all nodes, bandwidth-limited channel, unreliable wireless transmission medium, etc. This environment will surely induce a high delay variation and packet loss rate impairing dramatically the user experienced quality of conversational services such as VoIP. Indeed, such services require the reception of each media unit before its deadline to guarantee a synchronous playback process. This requirement is typically achieved by artificially delaying received packets inside a de-jitter buffer. To enhance the perceptual quality the buffering delay should be adjusted dynamically throughout the vocal conversation.

In this work, we describe the design of a playout algorithm tailored for real-time, packet-based voice conversations delivered over multi-hop wireless ad-hoc networks. The designed playout algorithm, which is denoted MAPA (mobility aware playout algorithm), adjusts the playout delay according to node mobility, which characterizes mobile ad-hoc networks, and talk-spurt, which is an intrinsic feature of voice signals. The detection of mobility is done in service passively at the receiver using several metrics gathered at the application layer. The perceptual quality is estimated using an augmented assessment approach relying on the ITU-T E-Model paradigm while including the time varying impairments observed by users throughout a packet-based voice conversation. Simulation results show that the tailored playout algorithm significantly outperforms conventional playout algorithms, specifically over a MANET with a high degree of mobility.

1. Introduction
A mobile ad-hoc network is a collection of heterogeneous nodes, which move randomly within an area and communicate with each other via wireless interfaces. Contrary to conventional last-hop wireless networks, MANETs enable the establishment of multi-hop wireless connections between any source–destination pair without the use of any existing infrastructure. In fact, inside a MANET all nodes act as routers and forward received packets to nodes within radio range.

MANETs are challenging networks since they exhibit several characteristics not found in conventional packet-based networks. For instance, the shared nature of the wireless channel entails an end-to-end effective data rate of few kilobits, despite the quick rise of point-to-point data rate which reaches currently 54 Mbps for some wireless cards and norms such as IEEE 802.11g and IEEE 802.11a [1]. This is mainly due to the interference problem between Intra and Inter flows in the same vicinity while crossing multiple wireless hops. In fact, Chen et al. proved that the maximum spatial reuse is roughly equal to 1/4 of
In this paper, we describe the design of a tailored voice packet playout algorithm, denoted MAPA, to suit network dynamics incurred by voice packets over a MANET. To this end, a comprehensive analysis of the key parameters of a voice conversation, namely one-way network delay and packet losses, is performed in order to adequately characterize the effect of mobility at the application layer. The designed algorithm MAPA adjusts dynamically the playout latency of received voice packets according to mobility-induced path switching, which is the main feature of MANETs, and talk-spurt, which is an intrinsic characteristic of voice signals. During path switching periods, voice packets are played cleverly on a per-packet basis while maximizing the perceptual quality as much as possible. However, during normal periods, MAPA plays voice packets according to a baseline per-talk-spurt playout algorithm which is extended to reduce the distortion effects due to severe compression and expansion of original silence period duration. The mobility is detected passively at the application layer by the receiver using the inter-packet delay difference coupled with packet out-of-order ratio metrics. Mobility-induced path switching is declared once measured parameters exceed a set of empirically calibrated thresholds. The rating factor is used to evaluate the performance of MAPA. This metric is estimated objectively using an extended computational assessment algorithm based on the E-model approach.

The remainder of this paper is organised as follows. Section 2 gives an overview of conventional voice packet playout algorithms designed for voice conversations running over a wide area IP network. Section 3 provides an analysis of key parameters of voice conversations over a MANET. Section 4 describes the algorithm MAPA, designed specifically to play voice packets transferred over a MANET. Section 5 describes the approach integrated in MAPA to detect mobility at the application level. An extended methodology to assess the perceptual performance of MAPA over a connection with time varying impairment is given in Section 6. Performance results and discussion are given in Section 7. Finally, we conclude in Section 8.

2. Related work

Real-time packet-based voice conversations require bounded end-to-end delay and delay jitter. Fig. 1 shows the temporal constraints linking the sender, transfer, and playout processes of a typical packet-based voice conversation transferred over a packet-based, best-effort network. The form of the staircase is due to the transmission of one packet every 20 ms, each of length 160 bytes. The generated traffic imitates the ITU-T G.711 codec output while disabling the voice activity detector algorithm, abbreviated as VAD. As shown in Fig. 1, voice packets reach the receiver side with variable network delays. The figure
illustrates two possible playout schedulers: the first corresponds to an adaptive playout algorithm, and the second corresponds to a static playout algorithm. Generally, adaptive algorithms provide better performance than static algorithms; however, adaptive approaches need more computation time than static approaches. By adopting a static playout strategy, the de-jitter buffer depth in packets \( L_b \) can be computed using the following equation:

\[ L_b = \left\lfloor \left( T_{\text{delay}}^1 - T_{\text{net}}^1 \right) \times C \right\rfloor + 1, \]

where \( T_{\text{delay}}^1 \) and \( T_{\text{net}}^1 \) represent respectively the playout and network delays of the first received packet and \( C \) corresponds to the data rate expressed in packets per second. By inspecting Eq. (1), we observe that the jitter buffer can compensate the effect of delay jitter as long as the experienced variation of one-way network delay between two consecutive packets is below \( L_b \times T_{\text{packet}} \), where \( T_{\text{packet}} \) is the packet duration in milliseconds. Cole and Rosenbluth studied in [6] some issues related to the performance of a static de-jitter buffer algorithm using a simplified probabilistic model. They derived a mathematical formula (see Eq. (2)) showing the relationship between the packet late ratio of a static de-jitter buffer algorithm, the buffer depth, and the mean and variance of inter-packet arrivals.

\[ e_{\text{de-jitter}} < v(j)/\left( g(b-1) \right)^2, \]

where \( j \) represents a random variable corresponding to the experienced inter-packet arrival interval to the de-jitter buffer, and, \( g \) and \( v(j) \) correspond respectively to its mean and variance, and \( b \) is the buffer depth in frames. This equation demystifies the trade-off between increasing the mouth-to-ear delay by incrementing the depth of the de-jitter buffer vs. decreasing the loss probability (see Fig. 2).

Typically, a dynamic de-jitter buffer algorithm is implemented at the receiver side to cope with varying jitter conditions. Several playout algorithms have been proposed in the literature to adjust the playout delay of a packet-based voice conversation to assure a user-friendly service. The major differences between the reported algorithms reside in the approach followed to predict the best playout delay to use during the next interval (or talk-spurt) and the amount of delay adjustment. As outlined previously, playout algorithms fall into two categories: network-based and application-based.

2.1. Network-based approaches

Network-based algorithms adapt the playout delay by monitoring some measured metrics at application layer such as the one-way network delay, the packet loss ratio, the late arrival ratio, the buffer depth, etc. This strategy considers only the real-time nature of the playout process. Playout algorithms that follow this approach may be fitted with minor adaptations for any conversational application (VoIP, video conference, etc.).

In an early work [7], we proposed an adjustment playout algorithm, denoted PAA (periodic adjustment algorithm), which adapts the playout delay periodically by monitoring the observed difference between the playout and network latencies. PAA was evaluated over wireless ad-hoc networks characterized by a high degree of mobility. The high mobility dictated the use of a small adaptation period in order to adjust the playout latency as quickly as possible to meet the network latency variations. Lui and Zarki proposed intra- and inter-flow synchronization controls for video conferencing running over IP networks [8]. Specifically, the authors aimed to synchronously play audio and video flows. The synchronization was done for each received flow and between concurrent flows [8]. They used a virtual clock at the receiver side initialized using the first received MDU (media data unit) and incremented at the frequency of the sender clock. Upon the arrival of each MDU, two measures were available: the generation time, which is measured at the source side and included in the packet header, and the arrival time measured according to the virtual clock. The occurrence of gaps can be easily detected by comparing the generation and arrival times. To adapt the playout instant, authors gathered – in a sliding window, having a maximum size of \( W_{\text{max}} \) – the recent observed values of network delays.
The playout delay adaptation mechanisms are triggered by two parameters: the synchronization phase distortion (SPD), denoted $\tau$, and the packet loss rate, denoted $l$, which are computed for each arrival media unit according to the current monitoring window. If $\tau$ or $l$ exceed their predetermined thresholds then the playout delay is adjusted by increasing the playout latency. In contrast, the authors propose to reduce the playout latency only when the measured synchronization errors for stream $i$ over the maximum monitoring window $W_i$ are all zero.

Stone and Jeffay proposed the QM (queue monitoring) algorithm which is a receiver-based playout algorithm relying on the monitoring of the depth of the playback audio/video buffer [9]. The idea behind QM stemmed from the fact that, when a significant and permanent reduction of network delay is observed, the de-jitter buffer depth will be increased for long duration. In such a case, the authors propose to reduce the playout delay by ignoring the oldest packet. To implement this strategy, QM uses arrays of counters and their associated thresholds to handle the playout queue. The receiver should execute the thresholding operation before playing each media packet and the oldest packet is suppressed from the queue when a counter exceeds its threshold, which is chosen empirically and retained fixed; the other counters are re-initialised to 0. The first threshold is set to infinity to prevent the underflow of the playout buffer.

2.2. Application-based approaches

Similarly to network-based playout algorithms, application-based playout algorithms adjust the playout delay according to a set of measured metrics (one way network delay, packet loss rate, etc.), however, they also consider the application features while adapting the playout latency. Playout algorithms for voice packets over IP networks represent a well-known solution which considers the voice signal features while adapting the playout latency. Since a received voice signal during a vocal conversation alternates between talk-spurt and silence periods [4,5], playout algorithms usually adapt the playout latency on a talk-spurt basis. The performance of per-talk-spurt playout algorithms are highly related to several configuration parameters. Indeed, the distribution of talk-spurts is critically, this will increase the collision probability between consecutive talk-spurts [4]. Formally, the collision problem occurs at the receiver when the playout time of the first packet of the $(k + 1)$th talk-spurt comes before the playout time of the last packet of the $k$th talk-spurt, causing an overlap between the beginning of the $(k + 1)$th talk-spurt and the end of the $k$th talk-spurt. This event should be avoided by the playout algorithm by appropriately delaying the starting instant of the next talk-spurt.

Several per-talk-spurt playout algorithms of voice packets over IP networks have been reported in the literature [4,5]. These algorithms compute the playout instant of the first packet of the next talk-spurt using Eq. (3). Then, subsequent packets are played at the instants given in the following equation:

\[
T^p_i = T^s_i + \bar{T}_{net} + \beta \times \psi_i, \quad (3)
\]

\[
T^p_i = T^s_i + (\bar{T}_1 - \bar{T}_s), \quad (4)
\]

where $T^p_i$ and $T^s_i$ represent, respectively, the playout and sending time of $i$th packet, and $\bar{T}_{net}$ is the weighted average of network delay upon the arrival of $i$th packet. $\psi_i$ is the mean delay variation which is updated for each received packet and $\beta$ is a coefficient to control delay/loss trade-off. Indeed, increasing the value of $\beta$ will result in the reduction of late packet arrivals at the expense of an increase of playout delay and vice versa. The recommended value of $\beta$, which is adopted in most current implementations, is equal to 4 [5]. As we can see, the computation of the one-way network delay for each received voice packet requires that the sender and receiver clocks be synchronized. The required data to compute the playout instants such as sending time, arrival time, and talk-spurt occurrence can be extracted from the headers of received packets.

Four basic reactive algorithms are described and analyzed extensively in [5]. These algorithms use Eqs. (3) and (4) to compute the playout instant of each received packet. The key difference between the proposed algorithms lies in the method used to update the mean network delay. By probing network delay of received voice packets, delivered over wide area IP networks, Ramjee et al. [5] show that, sometimes packet delays exhibit a sharp spike-like increase, which cannot be predicted in advance. After the occurrence of a spike, packets are received at a high frequency. Soon afterwards, the jitter returns to the normal state. This observation has lead to the design of an enhanced playout algorithm to update more adequately the statistical metrics used to track the network delays in order to effectively reduce the effect of delay jitters. The conceived algorithm adapts the weighted network delay according to the current state which can be Normal or Impulse. The start and end instants of a spike instance were detected by appropriately monitoring the incurred network delay variations using calibrated thresholds. All designed playout algorithms in [5] adapt the playout delay on per-talk-spurt basis which significantly restricts their adaptation capability. The assumption here is that the adaptation of playout delay within a talk-spurt dramatically deteriorates the perceptual quality which is true to a certain extent.

A natural extension of the previously described playout algorithms is proposed by Narbutt and Murphy [11], which...
consists of dynamically selecting the weighting coefficient \( z \) during a live packet-based voice conversation. This coefficient is used to update the mean and the variance of network delay. The authors build an off-line empirical function which associates with each lightly weighted delay variation the value of \( z \) that optimizes the trade-off between delay and loss. The lightly weighted delay variations are calculated using a small weighting factor of 0.6 to smooth out transit-delay variations. Generally, a high value is given to \( z \) when the playback process detects a low delay jitter inside the network. However, a low value is given to \( z \) when the playback process detects a high delay jitter inside the network. Performance evaluation of this extension shows that the adaptive playout algorithm with respect to \( z \) enables achieving a better loss-delay trade-off than conventional playout algorithms.

Recently, several emerging playout algorithms have been reported which aim at selecting the playout delay that maximizes the perceptual quality or minimizes the experienced impairments [12,13]. Fujimoto et al. proposed a quality-aware playout algorithm which selects the appropriate playout delay that maximizes the perceptual quality estimated using the mean opinion score, denoted MOS [12]. To this end, the authors use polynomial regression to derive a function to map objective measures, namely the playout delay and packet loss rate, to an opinion score. The resulting model requires measurement of the packet loss ratio due to network and the packet loss ratio due to late arrivals. To compute the packet late ratio, the authors used the fair assumption that one-way network delays over IP networks follow a Pareto distribution. Formally, the packet late ratio is given by:

\[
P_d = F(X > d) = \left( \frac{k}{d} \right)^z, \quad d \geq k,
\]

where \( d \) represents the playout delay used during the last talk-spurt, \( k \) and \( z \) are the distribution parameters given by:

\[
k = \min(d_1, d_2, \ldots, d_n) \quad \text{and} \quad z = n \left( \sum_{i=1}^{n} \log \left( \frac{d_i}{k} \right) \right)^{-1},
\]

where \( d_i \) represents the \( i \)th one way network delay stored in a histogram, logging the last 10,000 measured values.

In summary, adaptive playout algorithms with respect to the playout delay implicitly adapt the buffering delay by tracking the history of delay jitter. Naturally, the buffering latency is updated before each adaptation instance. To compute the buffer depth used in each period, we could re-calculate the buffer size using Eq. (1) upon the occurrence of an adjustment event. We note here, that any playout algorithm aims to appropriately select the buffer depth to conceal as much as possible the observed delay jitter. For planning purposes and when the playout algorithm is unspecified, we can fairly assume that the buffering delay is equal to:

\[
T_{de-jitter} = \min(T_i + 0.9 \times F_{RTT} + 300),
\]

where \( F_{RTT} \) represents the delay jitter computed according to RFC 1889, and \( T_i \) corresponds to the frame duration. In this equation all parameters are expressed in milliseconds. This value can be computed periodically to imitate the behavior of an adaptive playout algorithm.

We notice that the described playout algorithms were designed initially for wired networks characterized by relatively much smaller variations of end-to-end delay as well as a much lower packet loss ratio than for wireless mobile networks [4,5]. Indeed, the absence of node mobility in wired networks results in a relatively bounded delay jitter. These properties are invalid over multi-hop wireless networks where the delay variation and the packet loss rate are usually high. Moreover, recently optimized reported playout algorithms rely on the prediction of the optimal playout delay according to a specific processing of the one-way network delay history. Naturally, these approaches are unsuitable for voice conversations over a MANET where the delay history is useless due to network dynamics.

3. Analysis of one way delay and loss over MANETs

This section aims to reveal the effect of mobility on the key parameters of a voice conversations over a MANET, namely the one-way network delay and packet losses. As a baseline for comparison with the one way network delays experienced over a MANET, we plotted in Fig. 3a the stochastic portion of typical one way delays observed during a packet-based voice conversation over a wide area IP network. We note that conventional playout algorithms are designed to handle such delay variations. The plotted trace, collected using Nevot (network voice terminal), corresponds to the incurred one-way network delays of a packet based voice conversation established between two IP workstations deployed at the University of Massachusetts and in GMD FOKUS at the University of Berlin [4,14]. Fig. 3a illustrates a typical spike observed over IP networks spanning several talk-spurts. In fact, this particular observation led to the design of spike-aware playout algorithms. Fig. 3b shows the corresponding network delay distribution. As we can see, the one-way network delays can be conveniently modeled by a heavy tail distribution such as a Pareto distribution. The observed surges of network delays are mainly due to congestion problems inside the core routers. Indeed, router nodes in the Internet typically forward received packets according to a FIFO queuing strategy without any guarantees of end-to-end delay or packet loss rate.

The one-way network delay used as input parameter for any playout algorithm should be accurately measured. To this end, the sender and receiver clocks should be continuously synchronized. Time synchronization over wireless ad-hoc/sensor systems has been the subject of several recent studies [15]. Conventional synchronization protocols employed in wired networks are unsuitable for ad-hoc/sensor networks due to their computation and communication overhead. On-line clock synchronization in MANETs is outside the scope of this work. A suitable protocol should be deployed to synchronize the clocks of the wireless nodes. All our simulations assume that the time is set by a global clock and therefore clock skew is supposed zero.
To study the one-way network delays and packet loss of voice conversations over MANETs we resorted to simulation using NS2 [16]. In this study, we assume that nodes are moving according to Gauss–Markov (GM) mobility model. In contrast to the well-known random waypoint (RW) mobility model, the GM model achieves a gradual and smoother node movement [17]. Hence, movement patterns generated according to GM are more realistic than those obtained using RW. The traffic pattern should be carefully planned to satisfy the interactivity requirement. Further details are given in Section 7. Each simulated scenario involves 25 mobile nodes roaming freely, i.e., without obstacles, over a rectangular area of $900 \times 300$ m. The transmission range of each node is set to 250 m. These parameters result in a coverage area of 196,250 m$^2$, node density of one node per 10,800 m$^2$, a network diameter of 3.79 hops, and a mean network connectivity (node degree) of 18.17 nodes. These parameters avoid network partition occurrences with a high probability, which occurrences are not handled in this work. The wireless link bandwidth is 2 Mbps, and IEEE802.11b and dynamic source routing (DSR) are used as the MAC and routing layer protocols, respectively. The selection of DSR as routing protocol is based on earlier experiments where we found that DSR is the most suitable for conversational applications [18].

Five bidirectional voice conversions were established during the simulation run. This workload was selected after running several simulations having a different mobility scenario and load. In fact, analytically estimating the maximal workload, referred to as network capacity, of an arbitrary MANET is a challenge since the offered bandwidth is topology-related and time dependent [19]. Node velocities are randomly selected in the range $1-5$ m/s to mimic node mobility in an urban environment. Each voice conversation lasted for a randomly selected duration according to an exponential distribution with mean 3 min. The start time of each session was chosen uniformly. The simulation runs have been re-iterated using different mobility scenarios until all voice conversations incur a packet loss ratio under 15%. In our opinion, such a condition should be used as a baseline scenario for quality parameter analysis.

The selected studied scenario introduces a packet loss ratio varying between 0.88% and 10.32% for the generated voice conversations. The out-of-order packet ratio and mean network delay varied from 0% to 27.32% and from 15.55 ms to 583.946 ms, respectively. The mean path length of received packets varied between 1.453 hops and 5.226 hops. According to our analysis, the average path length of successfully received packets is equal to 3.33 hops. Further, we observed that some packets reached the receiver with a very high latency ($>5$ s). This is in large part due to the DSR routing protocol which follows a reactive approach where routes are built on demand. In fact, the discovery time to find an alternative path upon a path loss occurrence is topology-related. Moreover, the caching mechanism used by DSR increases the probability to receive out-of-order voice packets with high latency.

Fig. 4a shows the experienced one-way network delays over the simulated MANET during the voice conversation having the maximum packet loss ratio (session 9). As we can see, the network delay variations are very high and unpredictable. Moreover, in contrast to wide-area wired IP networks, where a spike decays linearly over time (see Fig. 3a), we observe high delay variations during the same delay spike (see Fig. 4b). In fact, contrary to wide-area cabled IP networks where delay variations are attributed mainly to congestion, network delay variations over a MANET may be attributed to using several forwarding paths between communicating parties throughout a voice session, implying the establishment of multi-hop connections characterized by varying properties such as the number of hops, the load of traversed links, the number of neighbors, etc. In Fig. 4c, we plotted the distribution of network delay observed over a MANET. This curve is built by averaging the distribution of each obtained delay trace involved in the simulated scenario. As we can observe, similar to cabled IP networks, the incurred delays have a very long tail. However, according to the plotted curve, an exponential distribution seems to be more suitable to model network delays over a MANET.

---

1 We observed that beyond five concurrent conversations, the packet loss ratio becomes excessively large (30%).
In Fig. 5, we plotted the cumulative distribution function (CDF) of experienced network delays to show the discrepancy between the voice packet delay distribution over a wide-area cabled IP network and a MANET. In fact, the CDF is important in order to know the experienced perceptual quality with reasonable accuracy. Indeed, the CDF can be used to compute the best playout delay to achieve the user-specified target loss ratio. This may be done off-line using delay traces or on-line using statistics gathered from live voice packets [4]. As expected, the CDF over a MANET is also characterized by a heavy tail. The CDF over a MANET plotted in Fig. 5 corresponds to the average CDF function of each voice packet stream involved in the simulated scenario. Fig. 5 shows that an important percentile of received packets arrives with a very low one-way network delay. This proves that, typically, communicating parties are close to each other. Moreover, we observe that the CDF of a cabled IP network increases gradually in contrast to that observed over a MANET. Further, we found that the perceptual quality over a mobile ad-hoc network differs significantly from one session to another and even from one flow to another belonging to the same session.

The restoration of a broken path may introduce a long blackout period which is awfully annoying to users. In such a case, voice packets can be lost either at the sender transmission buffer or at an intermediate node. In fact, in a MANET, such path blackout may last several seconds. This may be avoided by improving the routing or medium access protocols to consider the specificity of conversational applications. It seems that geographical routing is an excellent candidate to enable a quick, yet simple path restoration [20]. Further, intuitively, an intelligent topology control at MAC layer may reduce significantly the path loss occurrences. We notice that we assume that the simulated sender application is unable to adapt its transmission rate to network conditions. This is preferred for the applications compatibility in a large scale environment.

Fig. 6 shows the distribution of observed loss run lengths of all involved voice conversations (10 voice flows). A loss run of length k refers to a sequence of k consecutive lost packets [21]. It is well-verified empirically that burst packet losses more critically impair the perceptual quality than random losses [22]. In fact, losing four consecutive
packets will introduce annoying glitches to the clean voice sequence. In Fig. 7, we plotted the complementary cumulative distribution function (CCDF) of all involved sessions in the simulated scenario of packet loss run. As we can see, long-burst loss runs represent an important percentile of observed loss runs. According to Fig. 7, 43% of packet loss runs had a length greater than 10 packets. These results show that long burst losses are common over a MANET. Handling of path blackout is outside the scope of this work.

In summary, the high network delay variations over a MANET require adapting the conventional playout algorithms in order to reduce the effect of mobility at application layer. Conventional and recently proposed history-based playout algorithms are unable to select adequately the playout delay due to the network dynamics. In fact, it is preferable that the playout algorithm plays received packets on a per-packet basis without considering the history of delay variation once mobility is detected to achieve the highest quality.

4. Mobility aware playout algorithm (MAPA)

The diagram given in Fig. 8 summarizes how MAPA processes the received packets in order to compute their optimum playout instants. The algorithm MAPA is intended to be deployed on a wireless node characterized typically by its low processing capability. That is why MAPA is designed using event programming concepts. In this approach, a process waits passively (i.e., without the consumption of processor cycles) for a triggering event to do something. As shown in Fig. 8, the reception of a new voice packet is the event that triggers MAPA.

Two modes are defined to intelligently adapt the playout delay of played packets while minimizing the accompanied impairments of the perceptual quality. Specifically, MAPA operates in two possible modes: a Normal mode and a Smart mode. MAPA uses the Smart mode to compute the adequate playout delays at the occurrence of a path loss due to mobility. The occurrence of a mobility event is detected by the receiver through a set of measured metrics at the application level. The mobility detection mechanism will be described in more detail in the next section.

Initially, MAPA plays voice packets according to the playout delays calculated in Normal mode. At the occurrence of a mobility instance, MAPA switches from Normal to Smart mode. During Smart mode, the playout delay is calculated on a per-packet basis, while aiming to maximize the perceptual quality insofar as possible. This means that
the features of the voice signal are ignored once mobility is
detected. This choice is preferred to maximize the perceptual
guantity.

The actual state of the MAPA process is identified using
a global flag denoted MOB (see Fig. 9). This flag is set to
True once the MAPA process enters the Smart mode and
set to False when the MAPA process returns to Normal
mode. The flag MOB is maintained by the Mobility function
which is called at the reception of each new packet, regard-
less of whether we are in Smart or Normal mode. The start
and end of Smart mode is triggered by the Mobility func-
tion which will be detailed in the subsequent section.

As depicted in Fig. 9, in Normal mode, MAPA uses the
third reactive algorithm proposed by Ramjee et al. [5]. This
algorithm, denoted hereafter Algo.3, uses the minimum
one-way delay incurred during the last talk-spurt to com-
pute the playout delay to be used during the next talk-
spurt. Based on an earlier study of the three reactive
algorithms presented in [18], we found that Algo.3
achieves the best performance over a MANET. We have
improved Algo.3 to preserve as much as possible the origi-
inal structure of the played signal by avoiding severe com-
pression of silence periods. This permits avoiding the
eventual collision between consecutive talk-spurts, signif-
ically harming the perceptual quality. To this end, the ori-
ginal silence duration is computed by subtracting the
timestamps of the last received and played packets after
the occurrence of a new talk-spurt. Next, MAPA selects
the appropriate playout delay according to the allowed
compression ratio of silence duration, which can be param-
eterized by users, and the playout delay computed accord-
ing to Algo.3.

The selection of a reactive rather than a predictive play-
out algorithm is preferred for transmission over a MANET.
Indeed, it is obvious and verified empirically that a reactive
approach allows a faster adaptation of the playout delay to

---

**Algorithm**: Mobility Aware Playout Algorithm (MAPA)

Upon the reception of new packet

\textbf{IF} ( !$\text{MOB} $) \textbf{THEN} \\
$\text{MOB} = \text{Mobility}(i, T_p^i, T_i^i)$

/* Detection of Mobility */

\textbf{IF} ( $\text{MOB} $) \textbf{THEN}

$T_p = \text{SmartMode}(i, T_p^i, T_i^i)$

\textbf{ELSE}

/* Update network delay and variance where $\alpha$ is set to 0.99802 as
recommended */

$\hat{T}_{\text{net}}^i = \alpha \times \hat{T}_{\text{net}}^{i-1} + (1 - \alpha) \times T_{\text{net}}^i$

$\hat{v}^i = \alpha \times \hat{v}^{i-1} + (1 - \alpha) \times | T_{\text{net}}^i - \hat{T}_{\text{net}}^i |$

\textbf{IF} ( $T_{\text{net}}^i < T_{\text{min}}^i $) \textbf{THEN}

$T_{\text{min}} = T_{\text{net}}^i$

\textbf{END IF}

\textbf{IF} ( Marked = 1 ) \textbf{THEN}

/* Compute the playout instant of the first packet of new talkspurt while
considering collision phenomena */

$\text{SIL}_i = T_i^i - \hat{T}_{\text{net}}^{i-1}$

/* $\beta$ is set to 4 as recommended */

$T_p = T_{\text{min}} + \beta \times \hat{v}^i$

$T_p = \max( T_{\text{net}}^{i-1} + T_{\text{pack}} + \text{THR} \times \text{SIL}_i - \hat{T}_{\text{net}}^{i-1}, T_p )$

/* Re-initialize the minimal network delay to the first packet of next burst */

$T_{\text{min}} = T_i^i$

\textbf{ELSE}

/* use the last playout delay to read the packet */

$T_p = T_{p,i-1}$

\textbf{END IF}

\textbf{END IF}

\textbf{ELSE}

$T_p = \text{SmartMode}(i, T_p^i, T_i^i)$

$\text{MOB} = \text{Mobility}(i, T_p^i, T_i^i)$

\textbf{END IF}

---

Fig. 9. Pseudo code of mobility aware playout algorithm (MAPA).
track network delay variation than the predictive approach [10]. Moreover, in contrast to predictive algorithms, reactive algorithms do not require storing the history of observed values. This will considerably minimize the storage overhead. Further, predictive playout algorithms predict the optimal playout delay, which will be used during the next talk-spurt, according to a certain analysis of the delay history distribution. This approach is unsuitable for MANETs due to their network dynamics.

During Smart mode, MAPA plays received packets on a per-packet basis while avoiding the eventual collisions between consecutive talk-spurts. Fig. 10 presents the strategy performed by MAPA during Smart mode. Specifically, once a new voice packet is received, MAPA computes its playout instant according to the current playout delay. If the received voice packet corresponds to a late arrival than the playout instant is adjusted according to the arrival time and the last played-packet instant. Given the playout instants of last played and received packets, MAPA calculates the eventual gap which will be introduced in the rendered stream. According to the experiments performed by Hoene et al., ignoring a late packet introducing a gap duration smaller than 80 ms will result in better perceptual quality than playing it by increasing the playout latency [23]. Consequently, MAPA ignores a packet received late if the introduced gap is below 80 ms. In contrast, when the gap is above 80 ms, MAPA will play the late packet and the playout delay is increased consequently. MAPA considers the importance of distortion due to interactivity between communicating parties by ignoring any packet which will result in an increase of the playout delay above 400 ms, which may be tolerated over a MANET.

Hoene et al. recommend reducing the playout delay by compressing the next silence period in order to adapt si-

---

**Procedure: Smart Mode**

/* Verify if the received packet triggers the start of a new talk-spurt or no */
IF (!MARKED) THEN
/* Compute the possible scheduling instant */
    \[ T^i_p = T^i_s + T_p \]
IF \( (T^i_p < T^i_s) \) THEN
/* Late arrival */
    \[ T^i_p = \max(T^i_s, T^i_{p-1} + T_{pack} + T^i_s - T^i_{p-1}) \]
    \[ GAP = T^i_s - T^i_{p-1} \]
    IF \( (GAP < 80) \) THEN
/* If the introduced gap is below 80 ms then ignore the received packet and maintain the same playout delay */
    \[ T_p = T_{p-1} \]
ELSE
/* Playout the received packet by incrementing the playout delay without exceeding the nominal allowed end to end delay */
    \[ T_p = \min(0.4, T^i_p - T^i_s) \]
    INC = INC + T^i_p - T^i_s
END IF
/* Beginning of a new talk-spurt, decrease eventual increasing of playout delay while considering the collision */
ELSE
/* Compute the playout instant using the actual playout delay */
    \[ \hat{T}^i_p = T^i_s + T_p \]
/* Preserve the original structure of the rendered stream */
    \[ T_p = \max(T^i_{p-1} + T_{pack} + TH \times SIL, \hat{T}^i_p - INC) \]
/* Playout delay used for received packet */
    \[ T_p = T^i_p - T^i_s \]
/* Re-initialize the increasing amount of delay */
    INC = 0
END IF

---

Fig. 10. Pseudo code for computing of playout delay in Smart mode.
lently the playout delay, i.e., without impairing noticeably the perceived quality [23]. That is why MAPA tries to reduce the amount of introduced delay during the last talk-spurt in the next silence period. To this end, MAPA computes the amount of introduced delay which should be subtracted from the next silence period using a variable denoted INC (see Fig. 10). Similar to Normal mode, the adaptation of playout delay during Smart mode considers the maximal allowed compression ratio of a silence period. The increasing amount of delay is re-initialized after the adjustment of playout delay.

5. Mobility detection at the application level

Mobility-induced path switching is an intrinsic feature of a MANET. To successfully integrate conversational applications over a MANET, we should incorporate adequate mechanisms to handle mobility consequences such as delay jitters, packet disorder, and packet losses at sender and receiver sides. In fact, it is highly desirable to react in a user-defined manner to the effects of a mobility occurrence, i.e., users can be warned in advance using a user-perceived event such as a special beep or a luminous signal when conversational applications are unable to assure the service stability due to mobility. This requires reliably predicting and detecting mobility instances using sophisticated mechanisms. In the context of this work, we have enhanced the receiver behavior through the adaptation of a conventional smoothing buffer algorithm to reduce the perceptually annoying effect of large and unpredictable delay jitters, induced at application layer.

Basically, the mobility detection can be active or passive. Active approaches require the assistance of intermediate nodes to detect mobility instances. Typically, once mobility is detected at a forwarding node, a control message is generated and sent back toward the source. As we can deduce, in such scenario, the receiver node will be unconscious of path loss. As a consequence, the receiver node is unable to perform the required maintenance operations. It is possible to send a control message to the receiver node by modifying the behavior of intermediate nodes. This should be avoided due to compatibility problems in a large-scale environment. In contrast, passive approaches rely on a set of end-to-end measurements gathered at the receiver application or transport level. The source node may be informed of a path loss through feedback sent by the receiver node. As we can see, this strategy needs only to modify the behaviors of the sender and receiver nodes. This methodology has been adopted in this work.

Several objective measures may be gathered passively in service without intrusion at the application level of a receiver node. For instance, the receiver may measure the inter-packet delay difference, the short term throughput, and the packet out-of-order and loss ratio, denoted respectively IDD, STT, POR, and PLR [24]. These metrics should be accurately measured to precisely characterize the network state. This work aims to identify only a mobility-induced path switching state. Typically, during a path switching period, the receiver will experience a relatively high network delay variation coupled with a large out-of-order packet ratio. In fact, during the occurrence of path switching, packets reach the receiver in a definitely random manner. Typically, once a route is broken, packets within intermediate forwarding nodes are cached temporarily and delivered to receiver node at the nearest opportunity.

In summary, a route change occurrence is detected passively when the sink node detects a relatively high IDD value coupled with a high POR value. This heuristic seems to well suited to identify the network state. Indeed, in contrast to route switching state, a congestion state will induce high IDD values, but low POR values. IDD values are given by:

$$IDD = T_{s}^{i+1} - T_{a}^{i} - (T_{s}^{i+1} - T_{a}^{i}).$$

Typically, generated packets during conversational services are sent only once by the sender due to the interactivity requirement. As a consequence, the POR ratio may be calculated merely based on the sequence number due to the global order property. This information is included in the header of each sent packet. To do that, the receiver maintains a variable denoted $p_{\text{max}}$, representing the maximal sequence number received until now. Next, the occurrence of an out-of-order packet is detected once a packet reaches the source having a sequence number $p_{i}$ smaller than $p_{\text{max}}$. Otherwise, the value of $p_{i}$ is assigned to $p_{\text{max}}$. IDD and POR metrics are updated once a new voice packet reaches the receiver side. This is done whether the received packet is in order or not.

![Fig. 11.](image-url) Correlation between out of order packet ration and mobility induced path switching occurrence.
In order to show the high sensitivity of the POR metric to route-change occurrences, we plotted in Fig. 11a the number of out-of-order packets observed during a sampling period having a length of 0.5 s. In Fig. 11b, we plotted the instants when the source node sends the first route request once a destination unreachable occurrence is detected. Fig. 11 shows that the ratio of out-of-order packets is highly sensitive to route change occurrences. The delay trace used is selected from the set of trace files generated in Section 3. To increase the accuracy, we use, as in [24], the relative sample deviation (RSD) method to judge if a sampled value is HIGH or not.

RSD is a statistical algorithm used to judge the level of a sample value relative to the recent history of records [24]. The RSD algorithm returns a value lying between 0 and 1, which is used to classify the level of a sample as HIGH or not. RSD can be summarized as follows: assume sample values x vary in the range [0, R]. RSD divides the range R into N intervals I1, I2, ... In where the interval Ii holds sample values within [(i-1)R/N, iR/N]. Denote the total number of samples as S and the number of samples within interval Ii as s(Ii). Given x, its corresponding interval is Ii = x/R/N + 1. To decide how HIGH x is, RSD calculates the ratio of sample values below x over the total number of samples:

\[
\text{RSD}(x) = \frac{\sum_{i=1}^{x-1} s(I_i)}{S}. \tag{9}
\]

This corresponds to the CDF distribution at x. Given RSD(x), we can tell what percentage of sample values is lower than x. An RSD value close to one implies that x is HIGH with respect to the history records. To reflect the current network condition by RSD value, it is necessary to maintain an updated history record. To this end, a forgetting mechanism is applied as follows: after the reception of a new sample value x, we increment its corresponding counter s(Ix) by one, and calculate its RSD value. Next, we subtract dI proportionally from each interval counter s(Ii) so that \(\sum_{i=1}^{N} d_i = 1\) and \(d_i = s(I_i)/S\). After forgetting, the original sample distribution is maintained while the total number of samples S is kept constant. In this way, more weight is given to the new samples and exponentially less weight to older ones when calculating RSD.

6. Conversational quality assessment

It is highly desirable to evaluate the performance of conversational applications such as VoIP at the user level. The conversational quality of voice services is estimated through a rating factor, denoted R. The rating factor is a scalar ranging from 0 to 100 corresponding respectively to the worst and best transmission quality. Practically, a rating factor value smaller than 60 corresponds to an unsatisfactory transmission quality. ITU-T E-Model is a parametric computational algorithm enabling the objective derivation of the rating factor using a set of gathered measures throughout the mouth-to-ear (M2E) path [25]. The reduced formula to derive the adequate rating factor R assessing a VoIP conversation is given by [25]:

\[
R = 93.2 - I_d(T_a) - I_e(\text{CODEC, plr}) + A, \tag{10}
\]

where \(I_d\) models the impairments affecting the interactivity such as the absolute propagation delay and echoes, \(I_e\) models the impairments affecting the intelligibility of voice conversations such as low bit-rate CODEC and packet losses, and A represents an advantage factor that accounts for a user willingness to accept some quality degradation in return for ease of access (e.g., cell phone). \(T_a\) and plr correspond respectively to the mean absolute propagation delay and packet loss rate.

To use the ITU-T E-Model to assess live voice conversations in service without intrusion, analytical models of \(I_d\) and \(I_e\) should be developed and calibrated for each configuration. The distortion effects of \(I_d\) are well-known and easily modeled [6]. By assuming a perfect echo canceller, impairments contributed to \(I_d\) can be given by the following equation:

\[
I_d(T_a) = 0.024T_a + 0.11(T_a - 177.3)H(T_a - 177.3)
\]

where

\[
\begin{align*}
H(x) &= 1 & &\text{if } x < 0, \\
H(x) &= 0 & &\text{if } x \geq 0,
\end{align*}
\]

where \(T_a\) represents the mean absolute propagation delay including framing, buffering, and network delays. In contrast to \(I_d\), models of \(I_e\) should be developed and calibrated specifically for each CODEC. Typically, proposed models in the literature have the following form:

\[
I_e(\text{CODEC, plr}) = a + b \times \ln(1 + c \times \text{plr}), \tag{12}
\]

where plr corresponds to the end-to-end packet loss rate, and the constant coefficients \(a, b,\) and \(c\) are selected according to the behavior of each CODEC [6,13]. For instance, the adequate coefficients of the G.711 CODEC with packet loss concealment (PLC) capability are \(a = 0, b = 30,\) and \(c = 15\).

ITU-T E-Model should be appropriately extended to accurately estimate the conversational quality over time-varying impairment connections. A natural approach consists of dividing the conversation duration into fixed intervals which are assessed independently. Intuitively, it is preferred to select the same assessment interval duration used to derive analytical models in the order of 8–20 s. The produced rating factor is termed as “instantaneous rating factor”. Subjective experiments done by France Telecom proved that this approach exhibits a good correlation between objective and subjective measurements [22].

Apart from estimating the “instantaneous rating factor”, it is highly desirable to estimate the rating factor which users will give at the end of a voice conversation. To produce a reliable result, the recency effect should be considered. The recency effect is used to incorporate the relationship between the impairment location and the produced rating factor at the end of a voice conversation. To this end, we propose to calculate the rating factor including the recency effect, denoted \(RE\), using a weighted average of the produced “instantaneous rating factor” during a voice conversation. The weights are adequately selected to incorporate the recency effects. Formally, \(RE\) is given by:

\[
RE = \frac{\sum_{i=1}^{N} a_i R_i}{\sum_{i=1}^{N} a_i}, \tag{13}
\]
where \( N \) represents the number of assessed intervals and \( a_i \) corresponds to the weighting factor of the \( i \)th interval. The values of weighting factors increase toward the end of the assessed voice conversation. Specifically, weighting factors are given by:

\[
a_i = 0.0214 \times i^2 + 0.0214 \times i + 0.36,
\]

where \( i \) corresponds to the interval identifier. This function is derived using polynomial regression according to a set of weighting factors recommended in a similar ETSI instrumental (objective) assessment approach [26]. Fig. 12 shows the accuracy of the regression process applied on the original set of recommended values. In addition to the previously described recency effect, it has been shown subjectively that bad “instantaneous rating factors” located close to the end of a voice conversation have a stronger impact on the overall rating factor at the end of a voice conversation [26]. By considering this effect, the rating factor at the end of a conversation, \( R_{\text{end-of-call}} \), could be calculated as follows:

\[
R_{\text{end-of-call}} = R_{\text{RE}} - \frac{2}{N} \left( \bar{R} - \min_{i=1:N}(R_i) \right),
\]

where \( \bar{R} \) represents the nominal average value of “instantaneous rating factors”.

7. Performance evaluation

7.1. Simulation strategy

We used NS2 running on Linux to simulate packet-based voice conversations over a MANET [20]. To evaluate the performance of MAPA, we should definitely define the node mobility model and traffic pattern. In this work, we use BonnMotion free software to generate mobility traces according to the Gauss–Markov (GM) mobility models [17]. GM models node movement according to speed and direction values which are updated at discrete time intervals. The new values are randomly chosen from a normal distribution with the old value as a mean. This model avoids unnatural abrupt movement of nodes and assures a uniform distribution of nodes over the simulated area. The maximal speed is an external parameter specified by users. A static mobility scenario may be generated for maximal speed set to 0.

The traffic pattern should satisfy a set of constraints to represent realistic voice conversations. First, for each selected pair of nodes a bi-directional connection should be established. Second, interacting nodes are unable to communicate interactively with other nodes during an active voice conversation. This is done by dividing the set of nodes into two clusters. A voice conversation can only be established between two free nodes belonging to different clusters. The start time of a voice conversation is chosen uniformly. The session duration is selected according to an exponential distribution with mean 3 min. The generated voice packet stream of each session follows an ON/OFF model having as mean active and silence periods of 1.004 s and 1.386 s respectively. Packet size is set to 160 bytes (20 ms of active speech). This allows to mimic the ITU-T G.711 CODEC with VAD output accurately.

Pertinent network and scenario parameters are summarized in Table 1. During the evaluation stage, we only consider delay traces having a packet loss ratio below 10% in each direction of a voice session. To evaluate the performance of the tailored playout algorithm MAPA, we have designed and developed a playout buffer simulator. This simulator is currently included into EVOM software tool, which we designed to assess the quality of voice transmission over a MANET [18].

7.2. Results and discussion

In order to evaluate the performance of MAPA over a MANET, we generate four bidirectional voice conversations. This workload will result in a lightly loaded network. Therefore, delay jitters and losses are mainly due to node mobility. Naturally, history-based playout algorithms are unsuitable for mobile nodes in a MANET due to network dynamics. In fact, recently proposed playout algorithms are designed to appropriately process the delay history to estimate the most suitable playout delay to be used during the next talk-spurt. Therefore, we confine the evaluation of MAPA against the four important reactive playout algorithms proposed by Ramjee et al. [5]. In fact, there are several enhancements of baseline algorithms to cope with

![Fig. 12. Regressive function to obtain the corresponding weighting factor.](image-url)
delay jitters over a wide area IP network. Naturally, these enhancements should be reviewed as to suitability for MANETs.

Several parameters should be specified in order to accurately detect mobility events. Table 2 summarizes the selected parameters used during this series of experiments. These values have been calibrated empirically according to the features of processed traces. Moreover, the weight factor $\alpha$ is set to 0.99802 and the safety factor $\beta$ is set to 4. The silence compression threshold is set to 0.85.

Fig. 13a shows the behavior of Algo.1, Algo.2, Algo.3, Algo.4 as well as Algo.M corresponding respectively to first, second, third, and fourth playout algorithms described in [5] and MAPA, to track the one-way delay over a MANET. Introduced collisions by Algo.1, Algo.2, Algo.3, and Algo.4 are adequately handled by ignoring overlapped voice slots. The plotted one-way network delay curve corresponds to the network delays of a selected packet voice stream delivered over a simulated mobile network having a low degree of mobility (1 m/s) to mimic pedestrian motion. Following a careful analysis of results we found that high network delay variations during a spike entail a peculiar behavior of Algo.4, yielding a negative playout delay. This has been fixed by not updating the mean network delay when a negative value is detected. Moreover, we suspend the adaptive behavior of all playout algorithms when the resulting computed playout delay is greater than 400 ms. Fig. 13a shows that Algo.M tracks network delay variations more closely during a path switching occurrence. This is demonstrated in Fig. 13b which proves that Algo.M achieves the best trade-off between the ratio of late arrivals and average playout delay.

In order to show the behavior of MAPA at the perceptual level during a voice conversation over time varying impairment channels such as multi-hop connections, we plotted in Fig. 14 the estimated rating factor as a function of time using the instrumental assessment algorithm described in Section 6. The monitoring window used for quality assessment is set to 10 s. As shown in Fig. 14, MAPA clearly outperforms all other algorithms with respect to the instantaneous perceptual quality. Moreover, MAPA achieves the highest estimated perceptual quality at the end of the processed call.

In order to check the suitability of MAPA to cope with varying degrees of mobility, we plotted in Fig. 15 the average rating factor estimated at the end of each involved voice conversation (four voice flows) as a function of speed range, and compared it with the studied playout algorithms. The velocity of a node is chosen according to a user specified range varying from $V_{\text{min}}$ (minimal velocity) to $V_{\text{max}}$ (maximal velocity), with an increasing step of 5 m/s. According to the selected velocity, generated mobility scenarios may represent pedestrians or vehicle movement. Fig. 15 shows that algorithm MAPA outperforms conventional playout algorithms, especially over a MANET characterized by a low and moderate degree of mobility.
(<15 m/s). Recall that a voice conversation having a rating factor below 60 will result in unacceptable quality. According to Fig. 15, we notice that the four baseline playout algorithms will result in unacceptable quality when mobility is above 10 m/s. In contrast, algorithm MAPA assures an acceptable quality even for node speeds reaching 15 m/s. As we can see, for a high degree of mobility (>15 m/s), all playout algorithms achieve a poor perceptual quality. In our opinion, the perceptual quality in such circumstances may be improved by enhancing the performance of lower layer protocols.

8. Conclusion

In this article, we designed and developed a new playout algorithm, denoted MAPA, to play voice packets delivered over a MANET. A comprehensive study of key parameters of a voice conversation, namely one-way network delay and packet losses, is made to illustrate the discrepancy between the features of received packet voice stream over a wide area cabled IP network and a MANET. According to a set of extracted features, we designed MAPA to intelligently cope with mobility occurrence and talk spurts to achieve the highest quality over wireless ad-hoc networks. The detection of mobility is done passively at the application layer using the packet out-of-order ratio, coupled with inter-packet delay variation. The evaluation of MAPA with respect to perceptual level, using an augmented instrumental assessment algorithm, shows that MAPA performs at least as well as the four baseline playout algorithms for a low degree of mobility, whereas MAPA achieves a noticeable improvement of vocal service for a high degree of mobility. As future work, we plan to design on-line adaptation of the key parameters of MAPA, namely RSD thresholds, window size, and weighting factors. Further investigations regarding the performance of the end-to-end mobility detector and its improvement are solicited. Moreover, the source behavior should be sensitive to mobility occurrences. Further, we will investigate the possibility of improving the vocal perceptual quality by integrating QoS mechanism at the routing and link layers. In addition, the path blackout restoration problem for conversational applications will be addressed in more detail in future work.

References

[22] France Telecom R&D, Study of the relationship between instantaneous and overall subjective speech quality for time-


Sofiène Jelassi was born in Bizerte, Tunisia, in 1979. He received his Bachelor of Science and Master of Science degrees in Computer Science from Faculté des Sciences de Monastir (FSM), University of Monastir, Tunisia in 2003 and 2005, respectively. He is currently a Ph.D. student at the École Nationale des Sciences de l’Informatique (ESNI), University of Manouba, Tunisia. His research interests include ad-hoc wireless networks, heterogeneous networks, multimedia content delivering, conversational application integration, seamless mobility provision, and perceptual quality assessment.

Habib Youssef received a Diplôme d’Ingénieur en Informatique from the Faculté des Sciences de Tunis, University of El-Manar, Tunisia in June 1982 and a Ph.D. in computer science from the University of Minnesota, USA, in January 1990. From September 1990 to January 2001, he was a Faculty Member of the Computer Engineering Department of King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia (Assistant Professor from 1990 to 1995 and Associate Professor from September 1995 to January 2001). From February 2001 to June 2002, he was a Maître de Conférences en informatique at the Faculté des Sciences de Monastir (FSM), University of Monastir, Tunisia. From July 2002 to August 2005, he served as the Director of the Institut Supérieur d’Informatique et Mathématiques of the University of Monastir. He is currently serving as a Professor of computer science and Director of the Institut Supérieur d’Informatique et des Technologies de Télécoms, Hammam Sousse, University of Sousse, Tunisia. He has over 100 publications to his credit in the form of books, book chapters, and journal and conference papers. He is the author with S. Sait of two books: (1) “VLSI Physical Design Automation: Theory and Practice”, McGraw-Hill 1995 (also co-published by IEEE Press 1995), and reprinted with corrections by World Scientific in 1999; and (2) “Iterative Computer Algorithms with Applications in Engineering”, IEEE CS Press 1999, and since 2003 published by John Wiley & Sons, which has also been translated into Japanese. His main research interests are computer networks, performance evaluation of computer systems, and algorithms for the design automation of electronic systems.