Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 984923, 9 pages
doi:10.1155/2010/984923

Research Article

An Intelligent Packet Loss Control Heuristic for Connectionless Real-Time Voice Communication

Murat Şensoy,[1] Burcu Yılmaz,[2] and Erdoğan Yılmaz[3]

[1] *Department of Computing Science, University of Aberdeen, Aberdeen AB24 3FX, Scotland, UK*
[2] *Department of Computer Engineering, İstanbul Kültür University, 34156 İstanbul, Turkey*
[3] *Vocational School of Technical Sciences, İstanbul Kültür University, 34156 İstanbul, Turkey*

Correspondence should be addressed to Murat Şensoy, m.sensoy@abdn.ac.uk

Received 4 September 2009; Revised 1 April 2010; Accepted 21 April 2010

Academic Editor: Alexander P. Seyranian

Time critical nature of the real-time communication usually makes connection-oriented protocols such as TCP useless, because retransmission of old and probably expired packets is not desired. However, connectionless protocols such as UDP do not provide such packet loss control and suitable for real-time communication such as voice or video communication. In this paper, we present an adaptive approach for the intelligent packet loss control for connectionless real-time voice communication. Instead of detecting and resending lost voice packets, this heuristic estimates the packet loss rate adaptively using a modified version of reinforcement learning and resends the most critical packets before they are expired. Our simulations indicate that this approach is promising for a remarkable improvement in QoS of real-time voice communication.

1. Introduction

Today, real-time communication is getting more focus from both the academic community and industry. Unlike ordinary communications, real-time communication is highly sensitive to delays. In an ordinary data transmission, connection-oriented protocols such as TCP can be used to retransmit lost packets during the transmission. Design of TCP makes it retransmit lost packets until they reach their destinations. After a time of unsuccessful retransmission, TCP gives up retransmission of a lost packet. This time may be as long as 4 to 10 minutes depending on the implementation [1]. However, retransmissions of old and probably expired packets are useless when real-time communication is concerned. So, connectionless protocols providing no retransmissions of lost packets, such as UDP, are usually used in real-time communication such as voice or video communication. Using UDP may be reasonable when packet loss rates are low. However, it degrades QoS considerably while packet loss rate is increasing; the crucial packets may get lost during the communication without any change of determining and resending them.

In the literature, loss characteristics of communication channels are realistically modeled using Markov models [2, 3]. After determining the parameters of a model for a specific communication channel, we can use it to determine loss characteristics of the communication channel. That is, once we have a model of a communication channel, we can probabilistically estimate which UDP packets may arrive to the other side and which ones may get lost, with an uncertainty. Although there are various machine learning techniques that can be used to learn model parameters in various domains, Reinforcement Learning (RL) is well-tailored method to learn parameters of a Markov-based lossy channel models, because this learning technique is also based on a Markov decision process. Therefore, in this paper, we propose an adaptive algorithm depending on RL to estimate when to retransmit a packet without having information on whether it is lost or not. So, packets are retransmitted on the time before they are expired. This algorithm is designed to work with connectionless protocols such as RTP over UDP.

In the literature, RL is used to assist fulfillment of networking task in changing environments. Ferra et al. use RL for the scheduling of packets in routers [4]. In their study, there are different queues in the routers and each queue has a different QoS requirements. They use reinforcement learning to schedule packets in the queues so that quality constraint in terms of delay for each queue is attained. Wolpert et al. and Boyan et al. propose RL-based solutions for the routing under changing network conditions. Wolpert et al. use collective intelligence to route Internet traffic by introducing RL-based agents on the routers [5, 6]. They show that at its best settings, their RL-based routing algorithm achieves throughputs up to three and one half time better than that of the standard Belman-Ford routing algorithm [6]. Boyan et al. introduce a well-defined routing algorithm called Q-Routing depending on Q-learning algorithm of RL. Q-Routing is an adaptive algorithm, which provides efficient routing under changing network parameters such as link cost [7]. Chang et al. propose a way of improving routing decision-making and node mobility control in the scope of mobilized ad hoc networks using an RL approach [8]. According to the best of our knowledge, there is no stochastic packet loss control approach for the connectionless transport protocols in the literature. So, this study constitutes an exploration of this concept.

2. Intelligent Packet Loss Control Heuristic

2.1. Problem Definition

During a real-time communication in a noisy communication channel, packet loss is usually not compensated through retransmissions if a connectionless transport protocol is used. Let a sender try to communicate through such a channel, which has a changing packet loss rate. Intelligent packet loss control problem is to make decision on when to resend data packets and which packet should be resent in order to increase QoS in a dynamic environment without having information on whether the packets to be resent is lost or not.

2.2. Packet Loss Models for the Communication Channel

There are several models for the loss characteristic of communication channels [9]. A simple model is known as *Memoryless Packet Loss Model*. This model is a very simple Bernoulli loss model, characterized by a single parameter, the loss rate r. In this model, each packet is lost with a probability r. This model cannot be used for the modeling of bursty packet

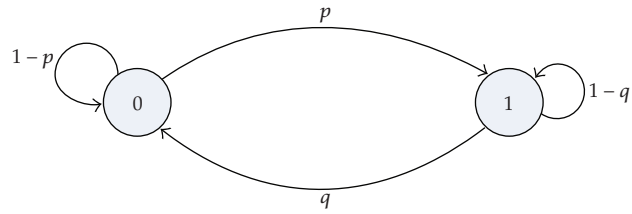

Figure 1: Bursty packet loss model which is also known as Gilbert model. The model is represented as a 2-state Markov chain. State 0 means that the current packet is not lost, whereas state 1 means the current packet is lost. In the model, p and q are probabilities.

loss. However, bursty packet loss is common during communications in packet-switched networks. Another model is *Bursty Packet Loss Model*, which is also known as Gilbert model. This model has been extensively used in the literature to model bursty packet loss in communication channels. Figure 1 demonstrates this simple model [3, 10]. It is defined by two states; state 0 means that the current packet is not lost whereas state 1 means that the packet is lost. Probabilities p and q define the loss characteristics of the channel. The probability q is related to the burstiness of the packet loss. That is, it defines the probability that the next packet is lost, provided that the previous one has arrived. Similarly, p defines the probability that the next packet arrives given that the previous one has been lost. According to the model, the average packet loss rate r is $p/(p+q)$ and the probability of getting a bursty packet loss of length n is $q \times (1-q)^{n-1}$. This model reduces to the Bernoulli model when the probabilities q and $1-q$ are equals. Gilbert's bursty packet loss model is used in this study in order to model the communication channels.

Unlike Bernoulli loss model, Gilbert's bursty packet loss model has a memory. The memory of a communication channel is defined as $\mu = 1-q-p$ [11]. When $\mu = 0$, the channel is memoryless; this means that the next state is independent of all previous states. However, if $\mu > 0$, the channel has a *persistent memory*, which means that the probability of remaining in a given state is higher than the steady-state probability of being in that state. On the other hand, if $\mu < 0$, the channel has an *oscillatory memory*, in which case the probability of staying in a specific state becomes lower than the steady-state probability of being in that state. A communication channel having an oscillatory memory would typically alternate frequently between State 0 and State 1, where as a communication channel having a persistent memory would typically stay for a long period in a state before alternating to another state. There are two extreme cases regarding the channel memory: (1) $\mu = +1$: in this case the channel remains forever in the initial state, (2) $\mu = -1$: the states alternate regularly. Therefore, we limit μ to the interval $[-1, +1]$ in our study to create a more realistic model of the lossy communication channels in real life.

2.3. Markov Decision Process

In order to model the sender's behavior in a lossy and lossless channel, a *Markov Decision Process* (MDP) is used [12]. Figure 2 shows the MDP of the sender. It is consists of two states. The first state is the Lossy state meaning that the communication channel used by the sender is lossy. The second state is the Lossless state meaning that the communication channel used by the sender is lossless. For lossless state, there is only one available action for sender agent. This action is Send action. The Send action meaning sending only the current packet to the

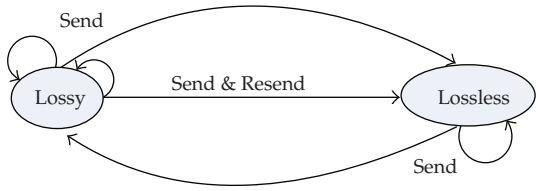

Figure 2: State diagram for the Markov Decision Process. There are two states, Lossy and Lossless. Lossless state has only one available action, Send. Lossy state has two available actions, Send and Send&Resend. Each action is indeterminist.

channel in a given time. On the other hand, there are two available actions for the Lossy state. The first action is Send and the second action is Send&Resend. The Send&Resend is an extension to the Send action. A sender agent sends the current packet and then tries to resend a previous packet at a given time if it chooses to execute Send&Resend action.

2.4. Reinforcement Learning Approach

The MDP depicted in Figure 2 shows the model of the environment with which sender agent interacts. In order to find utilities of choosing different action in a dynamic environment, reinforcement learning approach can be used. A modified version of Q-learning algorithm is used for this purpose. Q-learning algorithm is very sensitive to reward function. In order to get more realistic reward values for actions, time is divided into epochs. Each epoch has the same length and this length is measured in terms of the number of current packets sent. For example, if epoch length is 10, then each epoch takes sending 10 current packets. After each epoch, transmission statistics are sent to the sender agent by the receiver agent. During an epoch, only one action is executed in each time step by the sender agent.

2.4.1. Reward Function

An important issue while using RL is the definition of reward function. Rewards constitute the feedbacks to the actions of agents acting in a dynamic environment. So, definition of reward function is important. The reward function used in this study is shown in the following equation:

$$\text{reward} = \frac{\text{ArrivedSentPackets}}{\text{SentPackets}} + E(\text{RURP}) - \frac{\text{UnutilizedResentPackets}}{\text{ResentPackets}}. \tag{2.1}$$

In the equation, the ratio, ArrivedSentPackets/SentPackets, defines the ratio of arrived packets which are sent as current packets. This ratio is always the same for Send and Send&Resend actions. In the equation, the value of $E(\text{RURP})$ is the expected rate of unutilized resent packets. Those packets are the resent packets which are either lost in the channel or they are the copies of the previously arrived packets to the destination. The value of $E(\text{RURP})$ can be calculated using the simple statistics on recently transmitted packets. Let the expected packet loss rate estimated by the sender agent be r; then $E(\text{RURP})$ becomes $1 - r \times (1 - r)$ in which $r \times (1 - r)$ is the probability that the resent packet is successful arrived to the destination and it is not a copy of previously arrived packet. Lastly, in (2.1),

ratio UnutilizedResentPackets/ResentPackets represents the actual rate of unutilized resent packets in the previous epoch. According to (2.1), if utilization of resending packets decreases below the expected value then reward will decrease. So, Send action will be a better choice than Send&Resend action under those conditions.

2.4.2. Update of Q-Values

After defining the reward function for the problem, the next step is the update of Q-values. Q-values define the expected value of taking an action in a state. So, Q-values are crucial in terms of decision making on actions. Q-learning is a simple algorithm for the computation of Q-values [13, 14]. In this study, a modified version of Q-learning algorithm is used. Equation (2.2) shows the update of Q-values according to classical Q-learning algorithm. In the equation, s is the state before taking any action, $f(s)$ is the action, which is rational to execute in state s according to current policy, and s' is the resulting state after applying the chosen action. In Q-learning algorithm, $f(s)$ is the action with the highest Q-value among the actions available at state s. This deterministic nature of $f(s)$ makes Q-learning insensitive to changes in the environment. In order to handle this problem, sometimes actions other than $f(s)$ is chosen for exploration of the environment. This is an important issue in RL literature and known as *Exploitation versus Exploration* [13, 14]:

$$Q^{\text{new}}(s,a) = \alpha \times Q^{\text{old}}(s,a) + (1-\alpha) \times \left[\text{reward} + \gamma \times Q^{\text{old}}\left(s', f\left(s'\right)\right)\right]. \tag{2.2}$$

In order to embed exploration and exploitation into the formulation of the learning algorithm, actions are chosen probabilistically according to their Q-values. This means that the action with higher Q-value has the higher probability for being chosen. So, a modified version of (2.2) is used for the update of Q-values. In (2.2), $Q^{\text{old}}(s', f(s'))$ is replaced by the sum in (2.3). In (2.2), $(1-\alpha)$ is the learning rate and γ is discount factor, $0 < \gamma < 1$:

$$\sum_{a}\left[\Pr\left(s'\right) \times Q^{\text{old}}\left(s',a\right)\right], \quad \text{where } \Pr\left(s'\right) = \frac{Q^{\text{old}}(s',a)}{\sum_{b} Q^{\text{old}}(s',b)}. \tag{2.3}$$

2.5. Resending Previous Voice Packets

Resending is a part of Send&Resend action. A packet is chosen for resending according to combination of two factors: *Importance Criteria* and *Aging*. Characteristics of those factors change from application to application.

2.5.1. Importance Criteria

For some application, each packet has different degree of information and value. So, some packets may have higher degree of importance. For example, some speech segments are composed of a noise-like unvoiced speech signals and some speech segments composed of pseudoperiodic voiced speech signals. So, some speech packets may be very similar to the previously transmitted speech packets in a voice communication. This means that similarity to previous packets may be a measure for importance criterion in voice communication. In

this study, (2.4) is used for computation of I^t, the importance for a speech packet produced at time t. The equation compares a packet with previous m packets. In the equation, $E(\text{PLR})$ is expected packet loss rate and $S(t, t-i)$ is the similarity of the ith packet with respect to $(t-i)^{\text{th}}$ packet. In this study, a similarity function depending on the simple difference of *lpc* (Linear Predictive Coding [15]) parameters is used:

$$I^t = \sum_{i=1}^{m} E(\text{PLR})^i \times S(t, t-i). \tag{2.4}$$

In the equation, m depends on the bursty packet losses. Let the packet at $t-2$ reached to the destination. If the packet at time $t-1$ is lost, the packet at $t-2$ is copied instead of the packet at $t-1$ in the destination side. The probability of this case is $E(\text{PLR})$. While calculating the importance of the packet in time t without knowing which packets are lost and which ones are not, $S(t, t-2)$ must be weighted with $E(\text{PLR})$. This is the intuition behind (2.4).

2.5.2. Aging Function

Aging is an important criterion for the selection of the packets to be resent. For example, for a real-time communication session with rigid delay constraints, aging must be severe so that old and probably expired packets should not be resent. For different applications, different aging functions can be used. A general aging function used in this study is shown in (2.5). The equation shows the calculation of aging factor for the packet produced at time T (in the equation, t refers to the current time). In the equation, $1 \leq \text{agingBase}$. The parameter aginBase defines how much aging is important for a packet during the communication. Consider a voice mail application, where the delays in the voip packets are not important. In this case, aging of the delayed packets should be neglected by setting aginBase to one. On the other hand, in a real-time application with a certain QoS constraints, aginBase should be greater than one:

$$A^T = \text{agingBase}^{(t-T)}. \tag{2.5}$$

We should underline that the value of agingBase determines the importance of aging while selecting packets for resending. If last n packets are to be considered for resending, A and I parameters for each packet are multiplied and the packets are selected for resending probabilistically proportional to these multiplications. The intuition behind this weighting scheme (weighting importance with aging function) can be described as follows. Assume that the sender wants to resend a packet at time t_i. At this time, the last n packets are $p_1, p_2, \ldots, p_n$, where the oldest packet is p_1. After the sender sent a packet at time t_i, the last n packets will be $p_2, p_3, \ldots, p_{n+1}$. In this scenario, the packet p_1 can never be sent again to the other party if it is not sent at t_i. Therefore, older packets may be given more chance for resending, with respect to other packets with the same importance; otherwise these packets may never be sent to the other party in lossy channels.

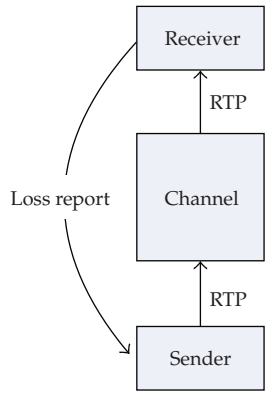

Figure 3: Simulation model for the intelligent packet loss control heuristic. There are three components: sender agent, communication channel, and receiver agent.

Note that, the values of n and agingBase depend on the application. For example, in a delay sensitive application, n should be small enough (e.g., 5), while agingBase should be set higher than 1.0 (e.g, 1.2).

3. Simulations

In order to evaluate the performance and abilities of the proposed intelligent packet control heuristic, several simulations are conducted. In each simulation, a voice communication is simulated. Figure 3 depicts the simulation model for the heuristic.

There are three basic components in the simulation model. First one is sender agent. Sender agent is the one implementing the intelligent packet control heuristic. It sends voice packets to the destination as RTP packets through communication channel using UDP as the underlying transport protocol. The sender agent uses the proposed RL approach with parameters $\alpha = 0.1$ and $\gamma = 0.8$ to compensate lost RTP packets by resending. The sender considers only last 5 packets for resending and uses agingBase = 1.2 while computing packet aging. Second component is communication channel. This channel is responsible for the transmission of the packets from sender to destination. Loss characteristics of the channel are modeled using Gilbert's bursty loss model. Packet loss rate should be change over time as in the real channels. So, parameters of Gilbert's model are changed over time in the simulations. That is, q and p have been varied within intervals $[0.9, 0.5]$ and $[0.4, 0]$, respectively, so bursty packet loss with different lengths has been guaranteed throughout the simulations ($-1 < \mu < +1$). The last component is receiver agent. Receiver agent receives packets from the channel and reports received packets to the sender using a reliable protocol such as TCP. Report interval is chosen as one epoch in the simulations. Length of one epoch is taken as 10 current packet transmissions.

Totally 10 voice communication sessions are simulated. During each session, the sender agent transmits the same voice stream to the destination. Loss rate of communication channel changes over time, but change of packet loss rates is the same for each simulation. So, mean performance of heuristic can be calculated using those 10 simulations for the same voice stream. Figure 4 shows the result of simulations. Before explaining the simulation results, there are some concepts requiring further explanation, such as *resending rate* and *loss compensation by resending*. Resending rate is simply the ratio of resent packets to the original

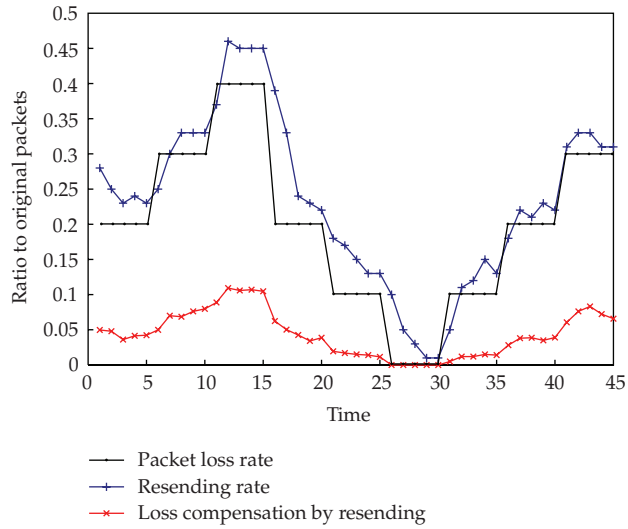

Figure 4: Simulation Results. The figure shows the mean values for "Resending Rate" and "Loss Compensation by Resending" for 10 simulations for the same voice stream. Change of Packet Loss Rates is same for each simulation.

packets. If there are 100 original packets and resending rate is 0.2 then total 120 packets are transmitted by the sender. Loss compensation by resending metric defines how successful the resending decisions are. Let there be 100 original packets but only 80 of them have arrived to the destination without resending. If resending of previous packets increases this number from 80 to 85 then *Loss Compensation by Resending* becomes $5/100 = 0.05$, which means that 5% of packets are compensated by resending.

As shown in the figure, throughout the communication sessions, bursty packet losses occur with different rates, because of the variations in the model parameters α and γ and the persistent memory of the communication channel (i.e., mostly $0 \leq \mu < 1.0$). Specifically, packet loss rate increases over time during the first 15 seconds, and then it decreases to zero by the 26th second. After a lossless period after 26th sec., the loss rates start increasing again until the end of the communication sessions. As a result of the burst packet losses in the channel, a considerable portion of the sent packets could not be reached to the receiver. The sender tries to compensate the loss of the packets by resending a limited number of previous packets as described in Section 2.

Sender agent uses the RL-based algorithm to make decisions of resending. So, changing packet loss rate is estimated and resending rate is approximated to the estimated packet loss rate. RL approach makes the agent choose alternative action to explore environment. So, there are some fluctuations in resending rate. However, resending rate is parallel to the packet loss rate in general depending on the choice of reward function. Due to the loss characteristics of the communication channel, only a portion of resent packets arrive to the destination. Also, some of the arrived resent packets are copies of previously arrived packets. So, only a portion of resent packets has utilized by the destination. This is shown by the curve for *Loss Compensation by Resending*. Although the resent packets utilized by the destination are low, those packets are the most important packets. However, selection of packets for resending is made depending on the importance criterion and aging. So, a few number of packets are expected to improve the voice quality considerably.

4. Conclusion and Future Work

In this study, an intelligent packet loss control heuristic for connectionless real-time voice communication is introduced. According to the best of our knowledge, this study is the first one as a stochastic packet loss control approach for the connectionless transport protocols in the literature. So, this study is a sort of introduction. The simulations conducted in this study show that an RL-based approach can successfully learn the loss characteristics of the communication channel. Resending packets without knowing which packets are lost decreases the utilization of resending but criteria for selection of packets to be resent increases the utility of resending. Only most important packets are resent. So, resending remains an important action. This study provides a novel tool for QoS improvement in connectionless real-time protocols such as RTP over UDP. However, this study does not provide objective and subjective performance analysis in terms of QoS and voice quality. This type of analysis is set a site as future work.

References

[1] W. R. Stevens, *UNIX Network Programming: Networking APIs: Sockets and XTI*, Prentice-Hall PTR, Upper Saddle River, NJ, USA, 1997.

[2] W. Jiang and H. Schulzrinne, "Modeling of packet loss and delay and their effect on real-time multimedia service quality," in *Proceedings of the Network and Operating System Support for Digital Audio and Video (NOSSDAV '00)*, pp. 1–10, 2000.

[3] H. Sanneck and G. Carle, "A framework model for packet loss metrics based on loss runlengths," in *Proceedings of SPIE/ACM SIGMM Multimedia Computing and Networking Conference*, pp. 177–187, 2000.

[4] H. L. Ferra, K. Lau, C. Leckie, and A. Tang, "Applying reinforcement learning to packet scheduling in routers," in *Proceedings of the 15th Conference on Innovative Applications of Artificial Intelligence*, pp. 79–84, 2003.

[5] D. H. Wolpert, K. Tumer, and J. Frank, "Using collective intelligence to route internet traffic," in *Proceedings of the Conference on Advances in Neural Information Processing Systems II*, pp. 952–958, 1999.

[6] D. H. Wolpert, S. Kirshner, C. J. Merz, and K. Tumer, "Adaptivity in agent-based routing for data networks," in *Proceedings of the 4th International Conference on Autonomous Agents (AGENTS '00)*, pp. 396–403, 2000.

[7] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: a reinforcement learning approach," in *Proceedings of the 24th Conference in Advances in Neural Information Processing Systems 6*, pp. 671–678, Morgan Kaufmann, 1994.

[8] Y.-H. Chang, T. Ho, and L. P. Kaelbling, "Mobilized ad-hoc networks: a reinforcement learning approach," in *Proceedings of the International Conference on Autonomic Computing*, pp. 240–247, 2004.

[9] G. Haßlinger and O. Hohlfeld, "The gilbert-elliott model for packet loss in real time services on the internet," in *Proceedings of the 14th GI/ITG Conference on Measurement, Modelling and Evaluation of Computer and Communication Systems (MMB '08)*, pp. 269–286, 2008.

[10] I. El Khayat, P. Geurts, and G. Leduc, "Enhancement of TCP over wired/wireless networks with packet loss classifiers inferred by supervised learning," *Wireless Networks*, vol. 16, no. 2, pp. 273–290, 2010.

[11] M. Mushkin and I. Bar-David, "Capacity and coding for the Gilbert-Elliot channels," *IEEE Transactions on Information Theory*, vol. 35, no. 6, pp. 1277–1290, 1989.

[12] M. Puterman, *Markov Decision Processes*, Wiley-Interscience, New York, NY, USA, 2005.

[13] D. H. Ballard, *An Introduction to Natural Computation*, MIT Press, Cambridge, Mass, USA, 1997.

[14] L. Kaelbling, M. Littman, and A. Moore, "Reinforcement learning: a survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[15] B. S. Atal, "Linear predictive coding of speech," *Computer Speech Processing*, pp. 81–124, 1985.