

PRESELECTION OF CANDIDATE UNITS IN A UNIT SELECTION-BASED TEXT-TO-SPEECH SYNTHESIS SYSTEM

Alistair Conkie Mark C. Beutnagel Ann K. Syrdal Philip E. Brown

AT&T Labs - Research, Florham Park, NJ, USA

ABSTRACT

Unit selection-based speech synthesis has recently been the focus of much attention in the speech synthesis community. In general, the speech quality from such a system achieves a high degree of naturalness and good intelligibility. However, examining and selecting units for synthesis as a runtime operation makes the unit selection process computationally expensive. Considerable attention has been focused on reducing the complexity of unit selection while maintaining quality.

Previous approaches to speeding up the process of runtime unit selection have focused on two aspects. (1) By limiting the number of candidate synthesis units considered in the unit selection process, the number of calculations required can be reduced. (2) By precomputing part of the needed calculations, the runtime complexity can be reduced. Much progress has been made using these methods, but usually at the expense of quality.

We present two methods of reducing the complexity of the calculation that avoid any reduction in synthesis quality, while allowing a very fast unit selection process. Results are presented for the reduction in complexity of the calculation process, and for a perceptual experiment that shows quality is not reduced relative to a full unit selection process.

1. INTRODUCTION

Unit selection-based speech synthesis[6],[9] has recently become the focus of much synthesis work. Unit selection is a method that seems to achieve more natural speech synthesis than the traditional diphone-based concatenative systems[11]. Until recently, the computational complexity associated with a unit selection approach made it unattractive or impractical. However, because computations can now be performed faster and data storage is cheaper, unit selection is being seen as an increasingly practical solution to the problem of achieving natural-sounding speech synthesis.

The high degree of naturalness and intelligibility it is possible to achieve with unit selection synthesis come with an

additional computation cost as compared to older systems. The overall computational cost is not necessarily different if one considers the complete process required to select diphones for the construction of a database, but in unit selection synthesis, the selection process is done automatically at runtime. This can be a very time-consuming operation, especially when a large database is used.

The selection process itself is done using a dynamic programming (Viterbi) algorithm. Costs are assigned to database phones that are candidates for a particular location in an utterance to be synthesized, and to the joins between individual units. A search is then performed to find the best sequence of candidates. The best candidates then get used in the synthesis procedure. The computational efficiency of this network and the degree to which it can be precomputed are key factors in determining the speed of a speech synthesis system based on unit selection.

Previous work on reducing the computational complexity of unit selection has focused on two areas. Some research[7],[10] has concentrated on finding ways to reduce the choice of candidates for synthesis by using decision tree methods. Other work[3] has tackled the problem of join cost calculations. A major concern was to maintain synthesis quality, as far as possible, while the computational complexity was being reduced. In the case of [3], a complexity reduction of at least a factor of four in the unit selection was reported, for no significant decrease in synthesis quality.

There are several other methods that can be used to minimize the time required to find suitable candidate units. In this paper we describe two methods of reducing the complexity of calculation and the effect that these methods have on speech quality and processing time.

2. The AT&T NEXT-GEN SYNTHESIS SYSTEM AND DATABASE

The AT&T Labs Next-Gen synthesizer is a hybrid system comprised of three parts. FlexTalk[1] provides the text analysis capabilities and produces a synthesis specification.

CHATR[4] unit selection has been adapted to provide sequences of units from our speech database that are then synthesized using PSOLA[11], HNM[12] or other waveform generation methods. The third component is Festival[5] from which we use the general architecture and some of the modules. The organizational structure is illustrated in Figure 1.

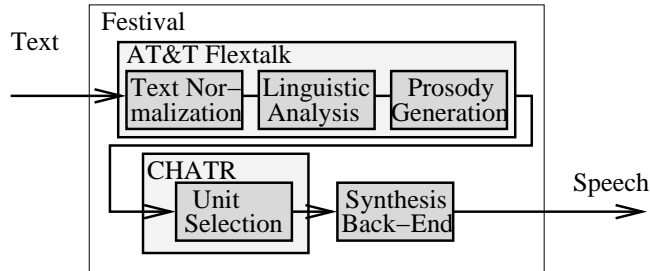


Figure 1: The Next-Gen TTS system

In the speech database there are normally many examples of each unit (in our case half phones). The database used for the work described here has 42,000 phones or 84,000 units. The most common half phone is represented ~ 3000 times in the database and the least common ~ 30 times. The data itself consists of recordings of a single speaker reading a variety of textual material, including newspaper text, interactive prompts and phonetically rich sentences.

3. UNIT SELECTION

The type of runtime unit selection used here was first described in the CHATR system[9]. A detailed description of the unit selection algorithm used in the AT&T Next Generation TTS system is given in [8]. Note again that the basic unit used is the half phone. The runtime unit selection uses *target costs* and *join costs*.

A *target cost* is associated with each unit in the database according to how well its specification matches the unit that we desire to synthesize. Only units with the specified phone identity are considered. Factors taken into consideration include F0, duration and elements of context. Formally, *target cost* T for a phone specification t_i and a database unit u_i is the weighted sum of target subcosts T_j , where p is the total number of individual features used in the specification, and w_j^t are the weights.

$$T(t_i, u_i) = \sum_{j=1}^p w_j^t T_j(t_i, u_i) \quad [1]$$

A *join cost* is associated with any two units that can join together for synthesis. It is defined as an acoustic cost, related to the spectral distance between elements on either side of a concatenation point. The cost is, however, not purely

acoustic, but may be modified according to context.

Formally, the *join cost* estimates the acoustic mismatch between two units. Join cost J for units u_{i-1} and u_i is the weighted sum of subcosts J_j , where q is the total number of acoustic features used, and w_j^c are the weights.

$$J(u_{i-1}, u_i) = \sum_{j=1}^q w_j^c J_j(u_{i-1}, u_i) \quad [2]$$

As described, a cost is associated with each possible unit and with all joins possible between units. We use these costs when we construct sequences of possible units for synthesis, by doing a Viterbi search.

The task of unit selection then is to find units u_i from the recorded inventory which minimize the sum of these two costs, accumulated across all phones i in the utterance:

$$C(t_i^n, u_i^n) = \sum_{i=1}^n T(t_i, u_i) + \sum_{i=2}^n J(u_{i-1}, u_i) \quad [3]$$

Since for our purposes a low cost indicates a good match with the specification, we seek to find a sequence of units with minimal total cost. The search network is represented graphically in figure 2.

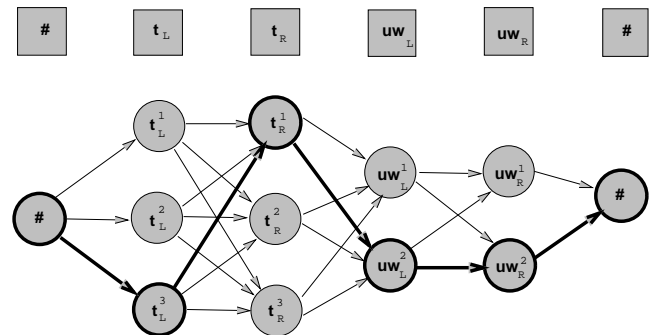


Figure 2: Viterbi search for the utterance “two”, with best path highlighted

4. PRESELECTION

The element of the runtime unit selection that performs calculations of target costs is divided into two parts for efficiency reasons. First, in a procedure termed *preselection*, a simple and fast cost calculation is performed over all the possible unit candidates, and the top n candidates are selected. Then for these n candidates a more complicated calculation is carried out. The initial (context-dependent) costs could be pretabulated, but currently storage requirements make this unattractive. The second phase is carried out with a more

expensive cost calculation that cannot (easily) be precomputed, because the cost depends on the parameters required to be synthesized (e.g. pitch, duration) and therefore is not known until runtime.

The default or baseline procedure is that each time a unit is specified in a synthesis sequence, every instance of that unit in the database is examined, and a *target cost* is associated with that instance. Then the phones surrounding the phone that we are trying to select are examined, and a cost is assigned to each candidate unit based on how appropriate it is likely to be for our purposes. The nearest neighbors have a greater influence in this process than the neighbors further away, reflecting the fact that the influence of a particular phone (for example via coarticulation) weakens with distance. Improvements to this baseline method are described in the following two sections.

5. PRESELECTION FILTERING

The first new method involves precomputing not the exact set of units that are relevant for each specific context, but a set of all the units that may be relevant for a closely related group of contexts. This set is in general much smaller than the set of all units of a particular unit type but, on average, is not greatly larger than the set of units that may be used for any one of the specific contexts. The unit selection procedure is carried out with this reduced list. This method has the additional advantage that it will produce output identical to the baseline system (with or without a join cost cache). Appreciable speedups in unit selection are achieved – a factor of two for one typical database we examined – with no loss of quality. If a particular sequence is not in the database, the system reverts to choosing the units from the set of all units of the type required according to the original system. This backup procedure is needed only infrequently. The dataset currently contains the most common 10,000 triphone sequences.

We consider a synthesis context of five phones, $p_i - p_j - p_k - p_l - p_m$, a center phone to be selected, p_k , and two context phones on either side. The purpose of the adjacent phones is to influence what candidate units get selected from the database to match p_k ¹. The first method takes account of the fact that we give higher weightings to the two inner context phones (p_j and p_l) than to the outer two phones (p_i and p_m). We calculate individual preselection costs for all database units of type p_k , given the sequence $p_i - p_j - p_k - p_l - p_m$. A maximum of n best units are selected. We denote the set of up to n units associated with this sequence by S_{ijklm} .

¹Note that in the case of half phones, units are chosen, in all cases, to be separated by one. For example, suppose the phone sequence to be synthesized is $k - ae - t$. Then in terms of half phones this would be $k1 - k2 - ae1 - ae2 - t1 - t2$. One sequence we would consider would be $k_1 - ae_1 - t_1$. The second would be $k_2 - ae_2 - t_2$. This is done to avoid including redundant information in the cost function (since the identity of one of the adjacent half phones is always known).

For the purposes of selecting units from the database that are possible candidate units for p_k , in the context $p_j - p_k - p_l$, we limit ourselves to using the set of database units S_{jkl} defined as follows:

$$S_{jkl} = \bigcup_{i \in P} \bigcup_{m \in P} S_{ijklm} \quad [4]$$

where P is the phone set. This list, S_{jkl} , is potentially very large and occasionally is the complete set of units of type p_k (i.e. no gain involved). Typically the list is rather close to the n units that get chosen for a particular 5-phone context, and substantially below the total number of units possible.

6. SYNTHESIS-BASED PRESELECTION

Beutnagel et al.[3] observed that nearly all the units in the database got used at some point (85%) and so pruning would be of limited value. This was validated, by experiment, in [2]. However, only a limited number of the potential joins ever got used. Based on this, Beutnagel et al. constructed a join cost cache that gave 98.2% of the units selected using the cache as equivalent to those selected in the original unit selection. Any join not in the cache was given a large fixed value. The join cost cache led to significant speedups in synthesis.

Our second new method of preselection does a more aggressive form of preselection by exploiting the fact that very few of the potential joins are ever used in practice. When compared with the first method, the second method achieves a considerable speedup and a considerable reduction in the size of the data files required. Synthesis-based preselection succeeds by allowing us to reduce drastically the candidate units that are considered at the preselection phase. This is done by examining the results of synthesizing a large number of sentences. For our experiments, synthesized sentences containing a total of 25 million phones were used, as described in [3]. From these data, the set of all triphone sequences $p_j - p_k - p_l$ that occur in the database was compiled (there are about 30,000 triphone sequences represented in our data, hence 60,000 half phone sequences). We tabulate the list of units that have been chosen for each context. Then, at synthesis time, rather than repeatedly examining the inventory of units afresh, we draw from the table of contexts of the list of units that have been used before in the given context, and these units become our candidate units for synthesis. Should a triphone sequence not be represented in the database (a rare but nevertheless possible case), the list of candidate units uses the baseline system in the same way as the first method.

7. RESULTS

This section presents results of the number of unit selection candidates tested and the number of join calculations

required in the baseline method, the preselection filtering method (Method 1) and the synthesis-based preselection method (Method 2). Additionally, a formal listening test was carried out and the results are presented below. All methods tested used half phones as units.

Table 1 gives a comparison of complexity for the baseline case and the two new preselection methods. A test data set was synthesized and critical parts of the unit selection were profiled. The results are summarized as two representative numbers, *candidates* and *joins*, as well as the overall speed of the system. The first count, *candidates*, tells us the number of calls to units in the database to find the preselection cost. This is a significant part of calculating the target cost. The second count, *joins*, tells us the number of times that a join cost is requested in the system. The number of calls in each case is given in millions. Note that a *candidate* calculation is approximately five times more expensive than a *join* calculation.

	Baseline	Method 1	Method 2
<i>candidates</i>	212M	59M	14M
<i>joins</i>	333M	325M	112M
Synthesis time	1.0	0.7	0.5

Table 1: A summary comparing pertinent data for baseline and the two new methods.

A formal listening test was conducted to compare subjective quality ratings of speech synthesized by the baseline and Method 2 TTS systems (Method 1 produces output identical to that from the baseline system and is not tested). Test stimuli consisted of three email messages that were synthesized by each of the two TTS systems tested. The stimuli were presented in random order, and listeners rated each one immediately after hearing it. Ratings were made on a standard 5-point scale (1=Bad, 2=Poor, 3=Fair, 4=Good, 5=Excellent). Forty-two normal-hearing adult native speakers of English served as listeners.

Mean Opinion Scores (MOS) for the two systems did not differ statistically from each other. The MOS for the baseline system was 3.58, and for the Method 2 system, 3.60. These results indicate that preselection did not degrade TTS quality when compared perceptually with full unit selection.

8. CONCLUSIONS

This paper has described two methods of reducing the complexity of unit selection in the the AT&T Next Gen speech synthesis system.

The first preselection method gives no decrease in quality for a substantial saving in terms of cost calculations.

The second preselection method does even better. The speed of unit selection was increased by a factor of 10 with no

significant loss in quality, as indicated by formal listening test results.

The methods described here, together with the work described in [3], yield a unit selection system (and overall TTS system) that efficiently produces high quality synthesis.

9. REFERENCES

1. Mark Beutnagel, Alistair Conkie, Juergen Schroeter, Yannis Stylianou, and Ann Syrdal. The AT&T next-gen TTS system. In *Proceedings of the 137th ASA meeting*, Berlin, 1999.
2. Mark Beutnagel, Alistair Conkie, and Ann K. Syrdal. Diphone synthesis using unit selection. In *Third International Workshop on Speech Synthesis*, Jenolan Caves, Australia, 1998.
3. Mark Beutnagel, Mehryar Mohri, and Michael Riley. Rapid unit selection from a large speech corpus for concatenative speech synthesis. In *Eurospeech*, Budapest, 1999.
4. A. Black. *CHATR, Version 0.8, a generic speech synthesis*. System documentation. ATR - Interpreting Telecommunications Laboratories, Kyoto, Japan, March 1996.
5. A. Black, P. Taylor, and R. Caley. The festival speech synthesis system. <http://www.cstr.ed.ac.uk/projects/festival.html>, 1998.
6. A. W. Black and N. Campbell. Optimizing selection of units from speech databases for concatenative synthesis. volume 1, pages 581–584, Madrid, Spain, 1995.
7. Alan W. Black and Paul Taylor. Automatically clustering similar units for unit selection in speech synthesis. volume 2, pages 601–604, Rhodes, Greece, 1997.
8. Alistair Conkie. Robust unit selection for speech synthesis. *Proc. 137th Meeting of the ASA*, 1999.
9. A. Hunt and A. Black. Unit selection in a concatenative speech synthesis system using a large speech database. *ICASSP*, 1:373–376, 1996.
10. M. W. Macon, A. E. Cronk, and J. Wouters. Generalization and discrimination in tree-structured unit selection. pages 195–200, November 1998.
11. E. Moulines and F. Charpentier. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9 (5/6):453–467, 1990.
12. J. Schroeter Y. Stylianou, T. Dutoit. Diphones concatenation using a harmonic plus noise model of speech. *Proc. EUROSPEECH*, Sept. 1997.