**Title:** SPEECH RECOGNIZER WITH A LEXICAL TREE BASED N-GRAM LANGUAGE MODEL

**Abstract:** In some embodiments, the invention includes a method comprising creating a lexical tree and identifying beginning phonemes in the lexical tree. The method of these embodiments further includes estimating probabilities of words in the lexical tree having particular ones of the beginning phonemes and storing at least some of the estimated probabilities, wherein backoff weights are not stored with the estimated probabilities. The estimated probabilities may be stored in a lookup table. In other embodiment, the invention includes a method of receiving phonemes and identifying them on a lexical tree. The method of these embodiments also includes estimating probabilities of words that include the phonemes through use of estimated probabilities retrieved from storage, wherein the retrieve probabilities do not include backoff weights stored with the estimated probabilities. Again, the estimated probabilities may be stored in a lookup table. The estimated probabilities may be used in establishing a pruning threshold. The methods may be implemented by instructions on a computer readable medium.
Published:
— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.
SPEECH RECOGNIZER WITH A LEXICAL TREE BASED N-GRAM LANGUAGE MODEL

Background of the Invention

Technical Field of the Invention: The present invention relates to speech recognition systems and, more particularly, to a lexical tree based n-gram language model.

Background Art: One component in a speech recognizer is the language model. Language models include probabilities that words in a vocabulary will occur and probabilities that a word will follow another word or more than one other word. Indeed, a popular way to capture the syntactic structure of a given language is using conditional probability to capture the sequential information embedded in the word strings in sentences. For example, if the current word is \( w_i \), a language model can be constructed indicating the probabilities that certain other words \( w_2, w_3, \ldots, w_n \), will follow \( w_i \). The conditional probabilities are typically calculated from examining the frequency of words following each other in a training corpus (e.g., newspapers). For example, the conditional probability \( p_{21} = p(w_2|w_1) \) is the probability that word \( w_2 \) will follow word \( w_1 \).

Probability \( p_{21} \) is referred to as a bi-gram. A tri-gram language model is the conditional probability that a word will follow two other words in order. For example, \( p_{310} = p(w_3|w_1, w_2) \) is the probability that word \( w_3 \) will follow word \( w_1 \), where \( w_1 \) follows word \( w_2 \). A uni-gram or 1-gram probability is merely the probability that a word will happen. For example, \( p_1 = p(w_1) \) is the probability that word \( w_1 \) will occur at a particular time, without consideration of a previous word.

The number of possible word combinations involved goes up exponentially from a uni-gram, bi-gram, tri-gram, etc. As used herein, the terms "lower gram" and "higher gram" refer to an order of grams. For example, a uni-gram is a lower gram than a bi-gram and a bi-gram is a lower gram than a tri-gram. A tri-gram is a higher gram than a bi-gram and a bi-gram is a higher gram than a uni-gram. For a large vocabulary, the total number of combinations of tri-grams and even bi-grams is unmanageably large. However, as it turns out, a great number of tri-grams and even bi-grams may have a conditional probability that is so small (or zero) that it is not worth including them in the language model. Backoff weights have been used to adjust the probabilities of
lower grams. For example, when an tri-gram probability is not included in the
language model, a bi-gram probability may be used when multiplied by a backoff
weight (bowt). If the backoff weight does not exist, the lower gram may be used in
place of the higher gram. Accordingly, a word based n-gram language model can be
represented as in equation (1) as follows:

\[
p(w_n | w_{n-1}, w_{n-2}, \cdots, w_i) = \begin{cases} 
  p(w_n | w_{n-1}, w_{n-2}, \cdots, w_i), & \text{if then-gram probability exists,} \\
  p(w_n | w_{n-1}, w_{n-2}, \cdots, w_i) \times \text{bowt}(w_1, w_2, \cdots, w_{i-1}), & \text{if the backoff weight exists,} \\
  p(w_n | w_{n-1}, w_{n-2}, \cdots, w_i), & \text{otherwise}
\end{cases}
\]

As noted above, even though equation (1), has the more general n-gram
presentation, it is rare to consider more than a tri-gram.

Typical n-gram language model file storage formats are illustrated follows:
for 1-grams: \( p(w_i) \) \( w_i \) bowt\( (w_i) \)
for i-grams: (for \( i = 1, \ldots, n-1 \)) \( p(w_i | w_{i-1}, \cdots, w_1) \) \( w_i \) \( \cdots \) \( w_i \) bowt\( (w_1, \cdots, w_{i-1}) \)
for n-grams: \( p(w_n | w_{n-1}, \cdots, w_i) \) \( w_i \) \( \cdots \) \( w_n \)

Lexical trees are used to organize possible words. For example, assume that in
a lexical tree, any one of words \( w_2, w_3, \ldots, w_N \) might follow word \( w_1 \). Conditional
probabilities can be computed to provide help in deciding which of words \( w_2, w_3, \ldots, w_N \)
follow word \( w_1 \). For a large vocabulary, the number of possibilities is enormous.
Techniques have been developed to reduce the number of probabilities involved
through using a “pruning beam” to “prune” away low probability paths having
conditional probabilities lower than a threshold with respect to the maximum.

The words are detected as a series of phonemes. As used herein, phonemes
refer to the digital electrical signals that represent acoustic sounds. However, it
typically is not known which word is being spoken until the last phoneme of the word
is detected, resulting in delayed pruning and overall slower decoding of received words.

The article, S. Ortmanns et al. “Language-Model Look-Ahead for Large
Vocabulary Speech Recognition,” ICSLP96 (1996), pp. 2095-98, proposed a lookahead
technique to incorporate language model probabilities earlier in the pruning process of
the beam search strategy. However, the authors of this article failed to appreciate how
to best keep stored estimated probabilities of the lexical tree to a manageable level. For
example, the Ortmanns et al. article concluded that the size of a table storing factored (estimated) probabilities would be prohibitively large. Id. at 2097.

Accordingly, there is a need for a better lexical tree n-gram language model format for a large vocabulary continuous speech recognizer (LVCSR).

Summary

In some embodiments, the invention includes a method comprising creating a lexical tree and identifying beginning phonemes in the lexical tree. The method of these embodiments further includes estimating probabilities of words in the lexical tree having particular ones of the beginning phonemes and storing at least some of the estimated probabilities, wherein backoff weights are not stored with the estimated probabilities. The estimated probabilities may be stored in a lookup table.

In other embodiment, the invention includes a method of receiving phonemes and identifying them on a lexical tree. The method of these embodiments also includes estimating probabilities of words that include the phonemes through use of estimated probabilities retrieved from storage, wherein the retrieve probabilities do not include backoff weights stored with the estimated probabilities. Again, the estimated probabilities may be stored in a lookup table.

The estimated probabilities may be used in establishing a pruning threshold.

The methods may be implemented by instructions on a computer readable medium.

Additional embodiments are described and claimed.

Brief Description of the Drawings

The invention will be understood more fully from the detailed description given below and from the accompanying drawings of embodiments of the invention which, however, should not be taken to limit the invention to the specific embodiments described, but are for explanation and understanding only.

FIG. 1 is a diagram representing a lexical tree according to some embodiments of the invention.

FIG. 2 is a high level schematic block diagram representation of a computer system that may be used in connection with some embodiments of the invention.
FIG. 3 is a high level schematic representation of a hand-held computer system that may be used in connection with some embodiments of the invention.

**Detailed Description**

The present invention involves a lexical tree based n-gram language model format for LVCSR. The invention allows estimating probabilities for a word once a beginning phoneme is detected. Pruning of paths that are below a threshold may begin before the successor word is identified. The invention is used to speed up the search process in a LVCSR. Language models play a key role for the decoding process, both in terms of accuracy and performance. Hence, speech recognition system performance relates to language models.

There are a variety of ways a lexical tree may be organized in connection with the invention. As one example, FIG. 1 illustrates portions of a representation of a lexical tree. The lexical tree of FIG. 1 joins words together according to phonemes and different words can share phonemes. A predecessor word w_0 is represented by a rectangle. There may or may not be a word before w_0. In the vocabulary, there are certain phonemes that can begin a successor word. Those beginning phonemes are Bp_h_1, Bp_h_2 ... Bp_h_x, which may be less than total number of phonemes.

Multiple words can begin with a phoneme. For convenience in discussion, words sharing phonemes have a similar labels. For example, words w_11, w_12, and w_13 each start with beginning phoneme Bp_h_1. More particularly, phonemes Bp_h_1, p_h_2, p_h_3, and p_h_4 make word w_11 (e.g., the word "fund"); phonemes Bp_h_1, p_h_2, p_h_3, p_h_4, and p_h_5 make word w_12 (e.g., "funds"), and phonemes Bp_h_1, and p_h_2 - p_h_4 and p_h_5 - p_h_10 make word w_13 (e.g., "fundamental"). (Note that the actual number of phonemes in the words may be different than that shown.) In practice, it would be typical for many more words to begin with the same phonemes, but for purposes of discussion, only three are shown in connection with Bp_h_1. In the example of FIG. 1, assume that word w_12 is the word finally detected. In that case, words w_0 and w_12 would be in the actual path, and other paths would be potential paths.

In some embodiments, once the first phoneme of a successor to word W_0 is identified, an estimate of the probability of the successor word is made so that pruning can begin before the successor word is known with certainty.
In some embodiments of the invention, a lexical tree based n-gram language model format is used which can be efficiently applied to the language model lookahead mechanism used in connection with, for example, a tree based Viterbi decoding algorithm. For a tree based Viterbi beam search algorithm, normally a estimated language model probability \( \pi_s(s) \) for a tree state \( s \) and a predecessor word string \( w_{n-1}w_{n-2}\cdots w_j \) can be estimated by equation (2) as follows:

\[
\pi_s(s) = \max_{w \in W(s)} (\lambda_w \cdot p(w \mid w_{n-1}w_{n-2}\cdots w_j))
\]  

(2)

where \( W(s) \) is the set of words that can be reached from a lexical tree state \( s \), \( \lambda_w \) denotes a fractional weight, \( v \) is the predecessor word, and \( p(w \mid w_{n-1}w_{n-2}\cdots w_j) \) denotes the n-gram conditional word probabilities. \( \pi_s(s) \) may also be called the estimated probability \( p_{\text{estimated}} \) used in establishing the pruning threshold. The estimated probability may be called the lookahead probability. As a result of applying language model lookahead, a much tighter pruning beam can be achieved to speed up the decoding process. The fractional weight \( \lambda_w \) can be set to 1 or can be between 0 and 1. In some embodiments, \( \lambda_w \) might be more than one. The fractional weight can be determined empirically, through trial and error, or calculated. The fractional weight may be the same or different for each Bph1. Although the invention is expressed in terms of n-grams, in actual implementations tri-grams, bi-grams, uni-grams and/or other grams may be used.

A tree state is from the perspective of a phoneme node. As spoken words progress and the more phonemes are detected deeper into the tree, the estimated probability may be recalculated so that pruning may continue.

Usually the above mentioned estimated (factored) language model probabilities have to be computed and generated on the fly during run time. This process can be time consuming even though a cache buffer is introduced to save the overall computational cost. Precomputing the estimated probabilities and storing them in a lookup table can significantly speed the process.

In the example of FIG. 1, assume that Bph1 is the first phoneme of the successor word. In this case, a bi-gram example of equation (2) is given in equation (3) as follows:
\[ p_{\text{estimated}} = \lambda_n \max \{ p(w_{11}|w_0), p(w_{12}|w_0), p(w_{13}|w_0) \} \]  

(3).

Words are pruned that have a probability or conditional probability that is below a threshold, or equal to or below the threshold, depending on the implementation. There are different ways in which the threshold can be derived. Examples include multiplying \( p_{\text{estimated}} \) by a number or subtracting a number from \( p_{\text{estimated}} \).

To speed up the decoding process, we define a lexical tree based n-gram language model format for storage of the pre-computed estimated probabilities by deploying the backoff mechanism to limit the memory requirement within a manageable range. The estimated probability \( p_{\text{estimated}} \) can be obtained in the general case as in equation (4) as follows:

\[
p_{\text{estimated}} = p(s_j | w_{n-1}, w_{n-2}, \ldots, w_1) = \begin{cases} 
\max_{w \in W(s)} (\lambda_n \cdot p(w | w_{n-1}, w_{n-2}, \ldots, w_1)), & \text{if word } n - \text{gram deriving it exists}, \\
p(s_j | w_{n-1}, w_{n-2}, \ldots, w_1) \times \text{bowt}(w_1 w_2 \ldots w_{n-1}), & \text{if backoff weight exists}, \\
p(s_j | w_{n-1}, w_{n-2}, \ldots, w_1), & \text{otherwise.}
\end{cases}
\]

where \( s_j \) is \( j \)th state of the potential successor word. Equation (4) includes three lines in the bracket. In the general case, the top line of equation (4) is equation (2). Of course, equation (4) can be used with different grams, such as uni-grams, bi-grams, and tri-grams. Equation (4) provides an approximation of equation (2). Only if the top line of equation (4) is satisfied are the \( p_{\text{estimated}} \) stored in storage, such as in a look-up table. In this way, the look-up table can be kept manageably small.

In equation (4), we do not need to store the backoff weights since they are identical to the weights stored in the standard word based n-gram language model. In decoding, the backoff weights can be obtained through a conventional file. In decoding if the first line of equation (4) is not met, the lower order estimated probability with a backoff weight if appropriate can be used.

The probability used for pruning can be merely the estimated probability of the successor word or the estimated probability added to the probability of the predecessor word (e.g., in FIG. 1, \( p(w_0) + p_{\text{estimated}} \)).

In some embodiments, the lookup table stores the tree based n-gram language model estimated probabilities as follows. However, other formats may be used.
1-grams:
\[ p(s_1) \quad s_1 \]

... 

i-grams: (for i=1, ..., n-1)
\[ p(s_i \mid w_{i-1} \cdots w_1) \quad w_i \cdots w_{i-1} \quad s_i \]

...

n-grams:
\[ p(s_n \mid w_{n-1} \cdots w_1) \quad w_i \cdots w_{n-1} \quad s_n \]

Since the total number of nodes from a compressed lexical tree is comparable to the total number of words in the lexicon, the total storage for a lexical tree based n-gram language model with the equation (4) approximation will be on the same order as compared to the corresponding conventional word based n-gram language model. The manipulation techniques used for a normal n-gram language model may be used on to the new lexical tree based language model file of the present invention.

In some embodiments, the estimated probabilities are calculated before recognition and stored in a lookup table. However, to reduce the size of the table, in some embodiments, only the entries which are derived directly from the n-gram probabilities (not by the backoff) are stored. The others which derived from the backoff probabilities (n-gram backoff to (n-1)-gram) are approximately backoffed to (n-1)-gram estimated probabilities. Through compression, the size of the table can be reduced to a manageable level.

When the last phoneme (or end node) of a word is reached, the successor word can be identified. For example, in FIG. 1, once phoneme \( ph_5 \) is reached, it is known that the word is \( w_{12} \). Once the word is known, the estimated probability can be replaced with an actual probability. This can be accomplished by adding the true conditional probability (e.g., in FIG. 1 \( p(w_{12} \mid w_6) \)) and subtracting the estimated probability. In some embodiments, the accumulated probabilities during a search may be started from the 1st word hypothesis, for example, \[ p(w_1, w_2, w_3 \ldots w_i) = p(w_1) + p(w_2 \mid w_1) + P(w_3 \mid w_2) + \ldots + p(w_i \mid w_{i-1}) \]. Log probabilities may be used such that addition is used for multiplication: \[ \log (p_1 \ast p_2) = \log(p_1) + \log(p_2) \].
The true probability, determined after the last phoneme is identified, may be expressed as \( p_{true} = p(w_{predecessor}) + p_{estimated} + p(w_{actual}|w_{predecessor}) \cdot p_{estimated} \). In the example of FIG. 1, assume word \( w_{12} \) is the actual word, the true probability \( p_{true} = p(w_0) + p_{estimated} + p(w_{12}|w_0) \cdot p_{estimated} \), where \( p_{estimated} \) may be obtained as described above.

The nodes of the lexical tree can be collapsed or compressed through eliminating redundant nodes. For example, in FIG. 1, phonemes \( \text{Bph}_1, \text{ph}_2, \text{ph}_3, \text{and ph}_4 \) could be collapsed into one state (node). In practice, however, there typically would be other words branching off of \( \text{Bph}_1 \), so it might not be collapsed with \( \text{ph}_2 - \text{ph}_4 \).

Phonemes \( \text{ph}_6 - \text{ph}_{10} \) might be collapsed into a state. In some embodiments, there are two lexical trees: an original one used for a speech recognizer and a compressed lexical tree for the language model. The compressed lexical tree may be used for creation of the look-up table during training. In training, lexical tree may be created from a lexicon according to known techniques.

There are a variety of computer systems that may be used in training and using a speech recognition system. Merely as an example, FIG. 2 illustrates a highly schematic representation of a computer system 10 which includes a processor 14, memory 16, and input/output and control block 18. There may be a substantially amount of memory in processor 14 and memory 16 may represent both memory that is off the chip of processor 14 or memory that is partially on and partially off the chip of processor 14. (Or memory 16 could be completely on the chip of processor 14). At least some of the input/output and control block 18 could be on the same chip as processor 14, or be on a separate chip. A microphone 26, monitor 30, additional memory 34, and input devices (such as a keyboard and mouse 38), a network connection 42, and speaker(s) 44 may interface with input/output and control block 18. Memory 34 represents a variety of memory such as a hard drive and CD ROM or DVD discs. A look-up table may take any of a variety of forms and is not intended to be a restrictive term. The stored estimated probabilities may be all in one place or spread around in different places. Part or all of the table may be copied and placed in different memories. A look-up table may be in memory 16, memory 34, or elsewhere. Look-up tables 22 and 24 represent all or part of the look-up table. It is emphasized that the system of FIG. 1 is merely exemplary and the invention is not limited to use with such a computer system.
Computer system 10 and other computer systems used to carry out the invention may be in a variety of forms, such as desktop, mainframe, and portable computers.

For example, FIG. 3 illustrates a handheld device 60, with a display 62, which may incorporate some or all the features of FIG. 2. The hand held device may at times interface with another computer system, such as that of FIG. 2. The shapes and relative sizes of the objects in FIG. 2 and 3 are not intended to suggest actual shapes and relative sizes.

The various memory may be considered computer readable mediums on which in instructions may be stored which when executed causes some embodiments of the invention to occur.

Other Information and Embodiments

A lexical tree based bi-gram language model with the above format has been implemented. By using pre-computed language model lookahead, we saved not only the computation cost for estimated probabilities that counts up to about 15% of the total computational time for our decoding task, but also about 50MB memory which required for the necessary cache buffer when generating those probabilities on the fly. (However, these numbers are merely examples, not requirements.) Moreover, our new language model format provides us the capability of handling higher order language model lookahead within reasonable time and memory.

Reference in the specification to "an embodiment," "one embodiment," "some embodiments," or "other embodiments" means that a particular feature, structure, or characteristic described in connection with the embodiments is included in at least some embodiments, but not necessarily all embodiments, of the invention. The various appearances "an embodiment," "one embodiment," or "some embodiments" are not necessarily all referring to the same embodiments.

If the specification states a component, feature, structure, or characteristic "may", "might", or "could" be included, that particular component, feature, structure, or characteristic is not required to be included. If the specification or claim refers to "a" or "an" element, that does not mean there is only one of the element. If the specification or claims refer to "an additional" element, that does not preclude there being more than one of the additional element.
Those skilled in the art having the benefit of this disclosure will appreciate that many other variations from the foregoing description and drawings may be made within the scope of the present invention. Accordingly, it is the following claims including any amendments thereto that define the scope of the invention.
CLAIMS

What is claimed is:

1. A method comprising:
   creating a lexical tree;
   identifying beginning phonemes in the lexical tree;
   estimating probabilities of words in the lexical tree having particular ones of the
   beginning phonemes; and
   storing at least some of the estimated probabilities, wherein backoff weights are
   not stored with the estimated probabilities.

2. The method of claim 1, wherein the estimated probabilities are stored
   only if corresponding n-grams exist.

3. The method of claim 1, wherein the estimated probabilities are stored in
   a look-up table.

4. The method of claim 3, wherein the look up table includes the following
   information:
   1-grams: \( p(s_i \mid s_j) \) \( s_i \)
   i-grams: (for \( i = 1, \ldots, n-1 \)): \( p(s_i \mid w_{i-1} \cdots w_i) \) \( w_i \cdots w_{i-1} \) \( s_i \)
   n-grams: \( p(s_n \mid w_{n-1} \cdots w_i) \) \( w_i \cdots w_{n-1} \) \( s_n \).

5. The method of claim 1, wherein the estimated probabilities \( p_{\text{estimated}} \) are
   created according to the following equation:

   \[
   p_{\text{estimated}} = \begin{cases} 
   \max_{w \in W(s)} (\lambda_w \cdot p(w \mid w_{n-1} \cdots w_i)), & \text{if word } n \text{-gram deriving it exists,} \\
   p(s_j \mid w_{n-1} w_{n-2} \cdots w_i) \times \text{bowt}(w_i w_2 \cdots w_{n-1}), & \text{if backoff weight exists,} \\
   p(s_j \mid w_{n-1} w_{n-2} \cdots w_i), & \text{otherwise.} 
   \end{cases}
   \]

   where \( s_j \) is \( j \)th state of a word associated with a particular one of the beginning
   phonemes, where \( W(s) \) is the set of words that can be reached from a lexical tree state
   \( s \), and \( \lambda_w \) denotes a fractional weight, and wherein the estimated probabilities are stored
   only if the first line of the above equation is met.

6. The method of claim 5, wherein \( \lambda_w \) is 1.
7. The method of claim 5, wherein \( \lambda_w \) is between 0 and 1 and is selected for each beginning phoneme.

8. A method comprising:
   receiving phonemes and identifying them on a lexical tree; and
   estimating probabilities of words that include the phonemes through use of
   estimated probabilities retrieved from storage, wherein the retrieve probabilities do not
   include backoff weights stored with the estimated probabilities.

9. The method of claim 8, wherein the estimated probabilities are stored in
   a lookup table.

10. The method of claim 9, wherein the lookup table includes the following
    information, wherein \( s \) is a state in the lexical tree, and \( p \) is a probability:
    
    1-grams: \( p(s_j) \quad s_i \)
    
    i-grams: (for \( i = 1, \ldots, n-1 \)): \( p(s_j \mid w_{i-1} \cdots w_i) \quad w_i \cdots w_{i-1} \quad s_i \)
    
    n-grams: \( p(s_n \mid w_{n-1} \cdots w_i) \quad w_i \cdots w_{n-1} \quad s_n \)

11. The method of claim 8, wherein backoff weight information can be
    derived from weights stored in a word based n-gram language model.

12. The method of claim 8, wherein the estimated probability is used in
    establishing a pruning threshold.

13. The method of claim 8, wherein the estimated probabilities are
    determined according to the following equation:

    \[
    P_{\text{estimated}} = \begin{cases} 
    \max_{w \in W(s)} (\lambda_w \cdot p(w \mid w_{n-1}w_{n-2}\cdots w_i)), & \text{if word } n-\text{gram deriving it exists}, \\
    p(s_j \mid w_{n-1}w_{n-2}\cdots w_i) \times \text{bowt}(w_iw_{i-1}\cdots w_{n-1}), & \text{if backoff weight exists}, \\
    p(s_j \mid w_{n-1}w_{n-2}\cdots w_2), & \text{otherwise}.
    \end{cases}
    \]

    where \( s_j \) is jth state of a word associated with a particular one of the beginning
    phonemes, where \( W(s) \) is the set of words that can be reached from a lexical tree state
    \( s \), and \( \lambda_w \) denotes a fractional weight, and only results of the first line are stored.
14. An apparatus comprising:

a computer readable medium having instructions thereon which when executed cause a computer system to:

creating a lexical tree;

identifying beginning phonemes in the lexical tree;
estimating probabilities of words in the lexical tree having particular ones of the beginning phonemes; and

storing at least some of the estimated probabilities, wherein backoff weights are not stored with the estimated probabilities.

15. The apparatus of claim 14, wherein the estimated probabilities are stored only if corresponding n-grams exist.

16. The apparatus of claim 14, wherein the estimated probabilities are stored in a look-up table.

17. The apparatus of claim 16, wherein the look up table includes the following information:

1-grams: \( p(s_i) \cdot s_i \)
i-grams: (for i=1, ..., n-1): \( p(s_i | w_{i-1} \cdots w_i) \cdot w_i \cdots w_{i-1} \cdot s_i \)
n-grams: \( p(s_n | w_{n-1} \cdots w_1) \cdot w_1 \cdots w_{n-1} \cdot s_n \).

18. The method of claim 14, wherein the estimated probabilities \( p_{\text{estimated}} \) are created according to the following equation:

\[
p_{\text{estimated}} = \left\{ \begin{array}{ll}
\max_{w \in W(s)} (\lambda \cdot p(w | w_{n-1} \cdots w_1)), & \text{if word } n \text{-gram deriving it exists,} \\
p(s_j | w_{n-1} \cdots w_2) \times \text{bowt}(w_1 w_2 \cdots w_{n-1}), & \text{if backoff weight exists,} \\
p(s_j | w_{n-1} w_{n-2} \cdots w_2), & \text{otherwise.}
\end{array} \right.
\]

where \( s_i \) is jth state of a word associated with a particular one of the beginning phonemes, where \( W(s) \) is the set of words that can be reached from a lexical tree state \( s \), and \( \lambda \) denotes a fractional weight, and wherein the estimated probabilities are stored only if the first line of the above equation is met.
19. An apparatus comprising:

a computer readable medium having instructions thereon which when executed
cause a computer system to:

receiving phonemes and identifying them on a lexical tree; and

estimating probabilities of words that include the phonemes through use of
estimated probability retrieved from storage, wherein the retrieve probabilities do not
include backoff weights stored with the estimated probabilities.

20. The method of claim 19, wherein the estimated probabilities are stored
in a lookup table.

21. The method of claim 20, wherein the look up table includes the
following information, wherein s is a state in the lexical tree, and p is a probability:

1-grams: \( p(s_i) \) \( s_i \)

i-grams: (for i=1, ..., n-1): \( p(s_i \mid w_{i-1} \cdots w_1) \) \( w_i \cdots w_{i-1} \) \( s_i \)

n-grams: \( p(s_n \mid w_{n-1} \cdots w_1) \) \( w_i \cdots w_{n-1} \) \( s_n \)

22. The method of claim 19, wherein backoff weight information can be
derived from weights stored in a word based n-gram language model.

23. The method of claim 19, wherein the estimated probabilities are
determined according to the following equation:

\[
P_{\text{estimated}} =
\begin{cases}
\max_{w \in \tilde{W}(s)} (\lambda_w \cdot p(w \mid w_{n-1} w_{n-2} \cdots w_1)), & \text{if word } n - \text{gram deriving it exists}, \\
p(s_j \mid w_{n-1} w_{n-2} \cdots w_2) \times \text{bowt}(w_2, w_{n-1}), & \text{if backoff weight exists}, \\
p(s_j \mid w_{n-1} w_{n-2} \cdots w_2), & \text{otherwise}.
\end{cases}
\]

where \( s_j \) is jth state of a word associated with a particular one of the beginning
phonemes, where \( \tilde{W}(s) \) is the set of words that can be reached from a lexical tree state
\( s \), and \( \lambda_w \) denotes a fractional weight, and results of the first line are precomputed and
stored.

24. The apparatus of claim 19, wherein the apparatus is a disk.
FIG. 1
FIG. 2

FIG. 3