

# Design and Implementation of Speech Recognition Systems

*Spring 2013*

Class 21: Subword units  
10 Apr 2013

# Topics for today

---

- The problem with word models
- Subword units
- Phonetic networks
- What is a good subword unit
- The effect of context
- Building word graphs
- Simplifying decoding structures
- Some speed-up issues
- Summary

# What does a Recognizer do

---

- Automatic speech recognition systems attempt to learn and identify **units of sound** in an utterance
- The “units” may be anything
  - They could be entire sentences, words or something finer
- To recognize a unit when it is spoken the system must learn models for it from training examples
  - We cover training in a later lecture
- The recognizer can recognize any of the units it has models for

# The problem with word models

---

- Word model based recognition:
  - Obtain a “template” or “model” for every word you want to recognize
    - And maybe for garbage
  - Recognize any given input data as being one of the *known* words
- Problem: We need to train models for every word we wish to recognize
  - E.g., if we have trained models for words “zero, one, .. nine”, and wish to add “oh” to the set, we must now learn a model for “oh”
- *Training needs data*
  - We can only learn models for words for which we have training data available

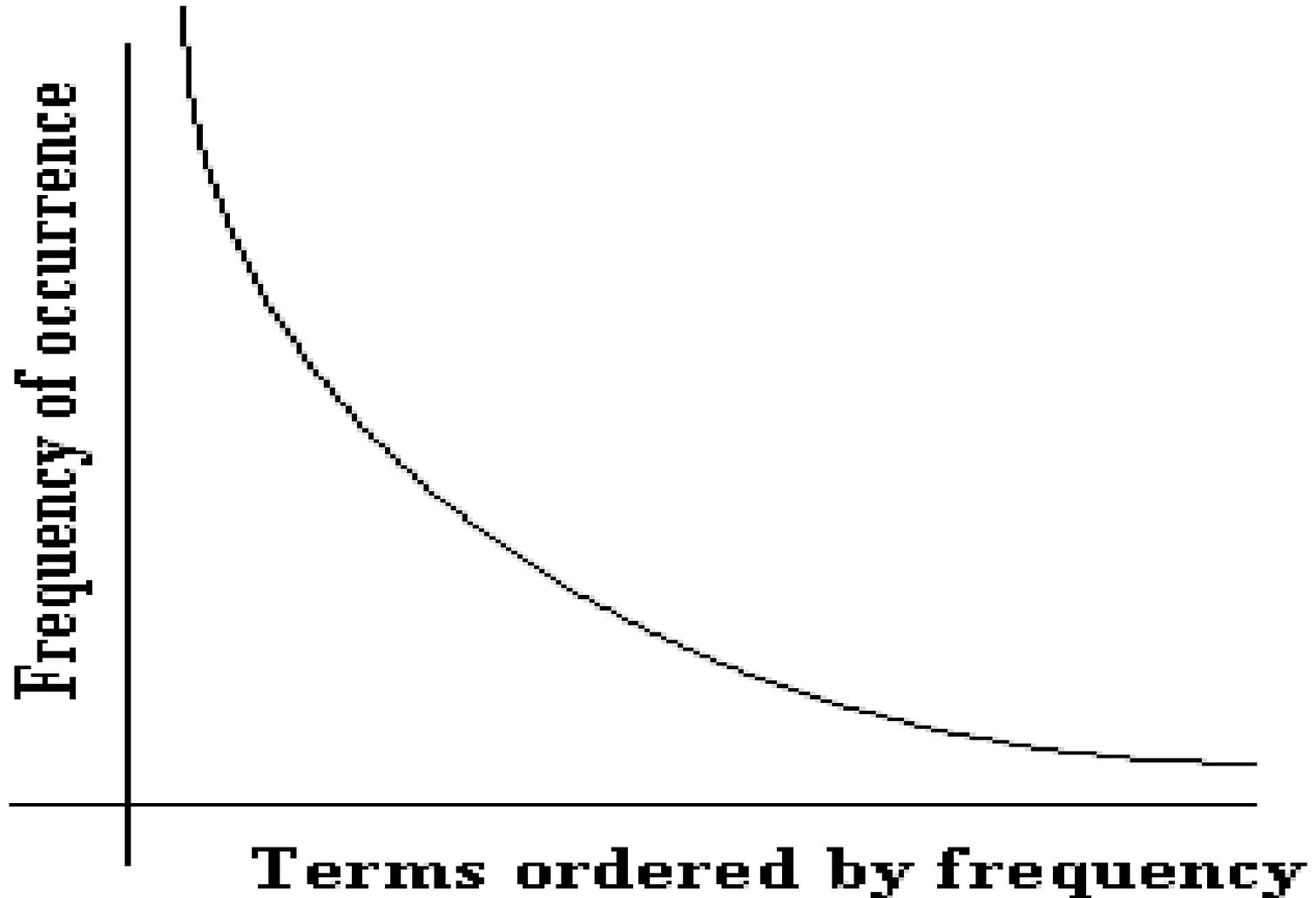
# Zipf's Law

---

- Zipf's law: The number of events that occur often is small, but the number of events that occur very rarely is very large.
  - E.g. you see a lot of dogs every day. There is one species of animal you see very often.
  - There are thousands of species of other animals you don't see except in a zoo. i.e. there are a very large number of species which you don't see often.
- If  $n$  represents the number of times an event occurs in a unit interval, the number of events that occur  $n$  times per unit time is proportional to  $1/n^\alpha$ , where  $\alpha$  is greater than 1
  - $\alpha > 1$

# Zipf's Law

---



# Zipf's Law also applies to Speech and Text

---

- The following are examples of the most frequent and the least frequent words in 1.5 million words of broadcast news representing 70 of hours of speech
  - THE: 81900
  - AND: 38000
  - A: 34200
  - TO: 31900
  - ..
  - ADVIL: 1
  - ZOOLOGY: 1
- Some words occur more than 10000 times (very frequent)
  - There are only a few such words: 16 in all
- Others occur only once or twice – 14900 words in all
  - Almost 50% of the vocabulary of this corpus
- The variation in number follows Zipf's law: there are a small number of frequent words, and a very large number of rare words
  - Unfortunately, the rare words are often the most important ones – the ones that carry the most information

# Word models for Large Vocabularies

---

- If we trained HMMs for individual words, most words would be trained on a small number (1-2) of instances (Zipf's law strikes)
  - The HMMs for these words would be poorly trained
  - The problem becomes more serious as the vocabulary size increases
- No HMMs can be trained for words that are never seen in the training corpus
- Direct training of word models is not an effective approach for large vocabulary speech recognition

# Sub-word Units

---

- Observation: Words in any language are formed by sequentially uttering a set of sounds
- The set of these sounds is small for any language
- We solve the problem by decomposing words into sub-word units
  - Units that are smaller than words, that must be concatenated to form words
  - Typically phonemes
- Any word in the language can be defined in terms of these units

# Phonemes and Dictionaries

---

- The mapping from words to phoneme sequences must be specified
  - Usually specified through a mapping table called a *dictionary*

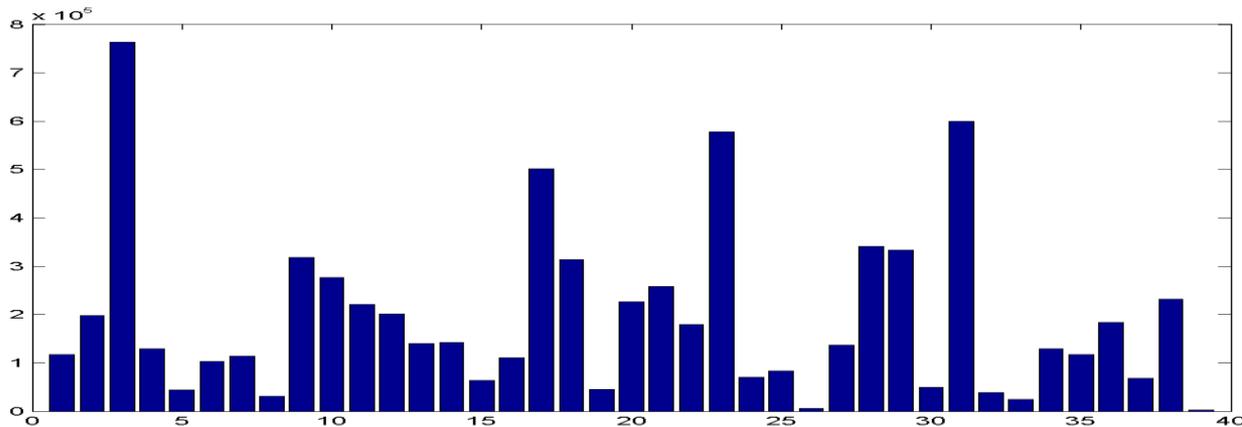
## Mapping table (dictionary)

<b>Eight</b>	ey	t			
<b>Four</b>	f	ow	r		
<b>One</b>	w	ax	n		
<b>Zero</b>	z	iy	r	ow	
<b>Five</b>	f	ay	v		
<b>Seven</b>	s	eh	v	ax	n

- Every word in the training corpus is converted to a sequence of phonemes
  - The transcripts for the training data effectively become sequences of phonemes
- HMMs are trained for the phonemes

# Beating Zipf's Law

- Distribution of phonemes in the BN corpus



Histogram of the number of occurrences of the 39 phonemes in 1.5 million words of Broadcast News

- There are far fewer “rare” phonemes, than words
  - This happens because the probability mass is distributed among fewer unique events
- If we train HMMs for phonemes instead of words, we will have enough data to train all HMMs

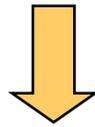
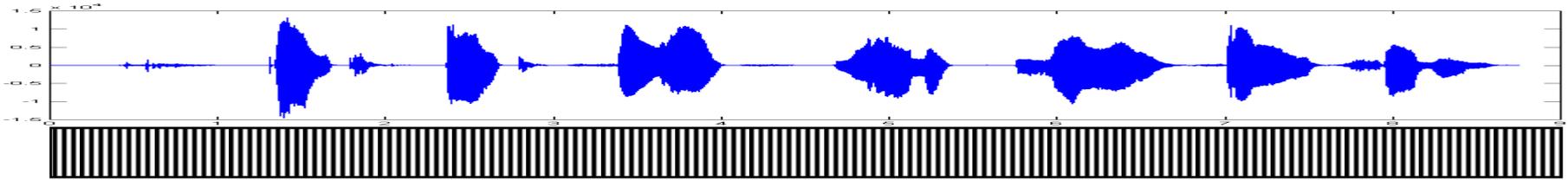
# But we want to recognize *Words*

---

- Recognition will *still* be performed over words
  - The Bayes' equation for speech recognition remains unmodified
- The HMMs for words are constructed by concatenating the HMMs for the individual phonemes within the word
  - In order to do this we need a phonetic breakup of words
  - This is provided by the dictionary
  - Since the component phoneme HMMs are well trained, the constructed word HMMs will also be well trained, even if the words are very rare in the training data
- This procedure has the advantage that we can now create word HMMs for words that were *never seen* in the acoustic model training data
  - We only need to know their pronunciation
  - Even the HMMs for these unseen (new) words will be well trained

# Word-based Recognition

Eight Eight Four One Zero Five Seven

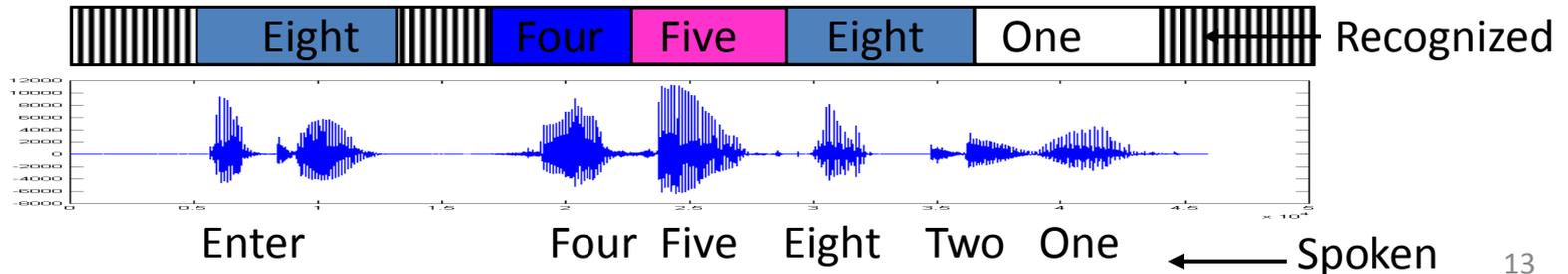
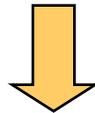


**Trainer**  
Learns characteristics  
of sound units

**Decoder**  
Identifies sound units based  
on learned characteristics

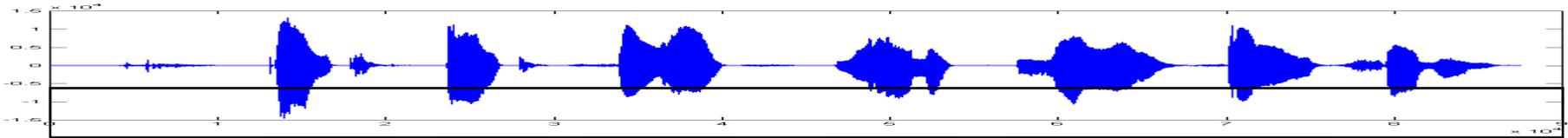
Word as unit

Insufficient data to train every word. Words not seen in training not recognized



# Phoneme based recognition

Eight Eight Four One Zero Five Seven



Eight Eight Four One Zero Five Seven

ey t ey t f ow r w a n z iy r o f ay v s e v e n

## Dictionary

Eight: ey t

Four: f ow r

One: w a n

Zero: z iy r ow

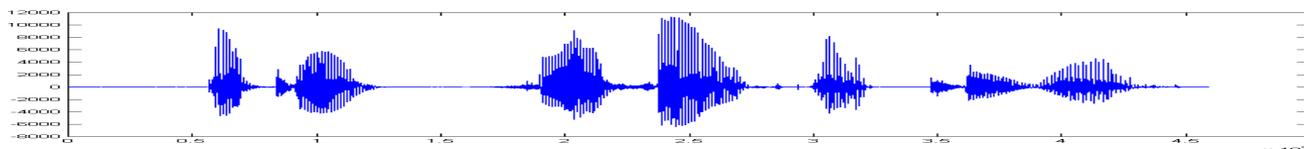
Five: f ay v

Seven: s e v e n

**Trainer**  
Learns characteristics  
of sound units

Map words into phoneme  
sequences

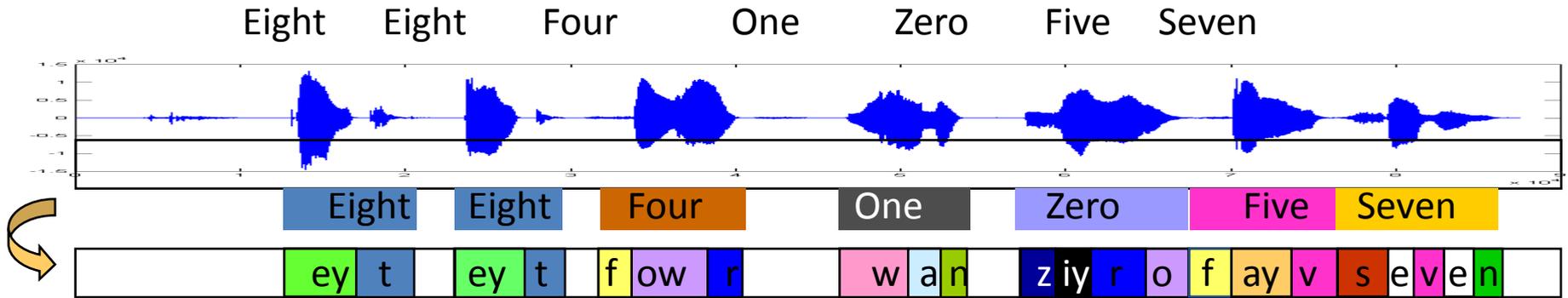
**Decoder**  
Identifies sound units based  
on learned characteristics



Enter

Four Five Eight Two One

# Phoneme based recognition



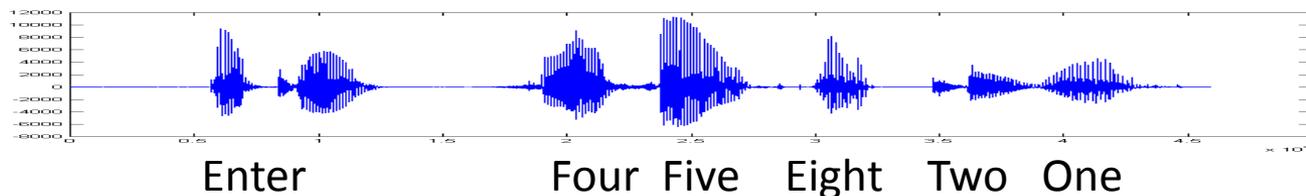
## Dictionary

Eight: ey t  
 Four: f ow r  
 One: w a n  
 Zero: z iy r ow  
 Five: f ay v  
 Seven: s e v e n  
 Enter: e n t e r  
 two: t u w

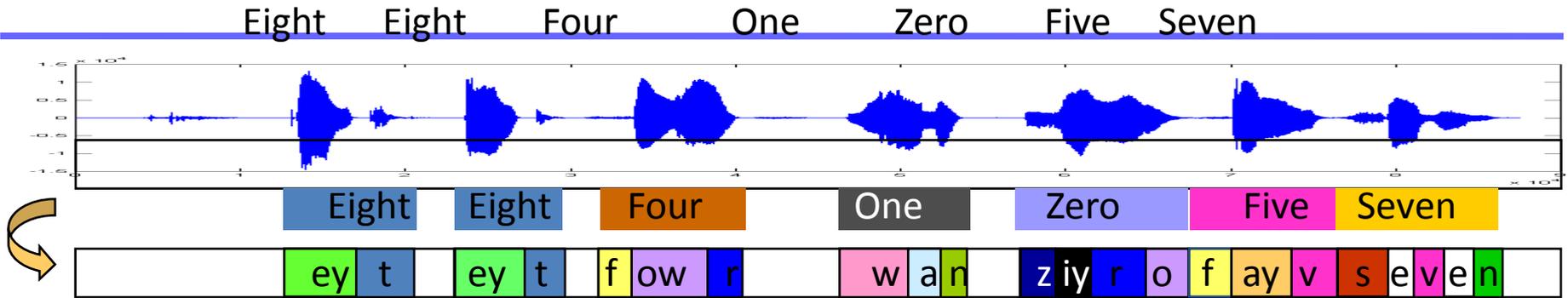
**Trainer**  
 Learns characteristics  
 of sound units

**Decoder**  
 Identifies sound units based  
 on learned characteristics

Map words into phoneme  
 sequences  
 and learn models for  
 phonemes  
 New words can be added  
 to the dictionary



# Phoneme based recognition



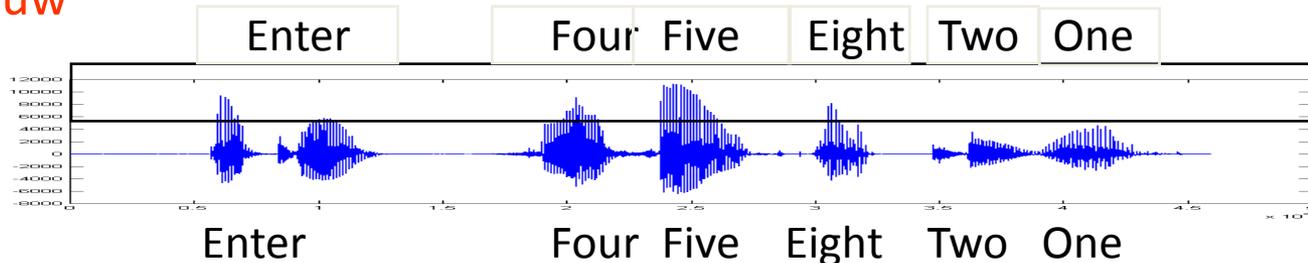
## Dictionary

Eight: ey t  
 Four: f ow r  
 One: w a n  
 Zero: z iy r ow  
 Five: f ay v  
 Seven: s e v e n  
 Enter: e n t e r  
 two: t u w

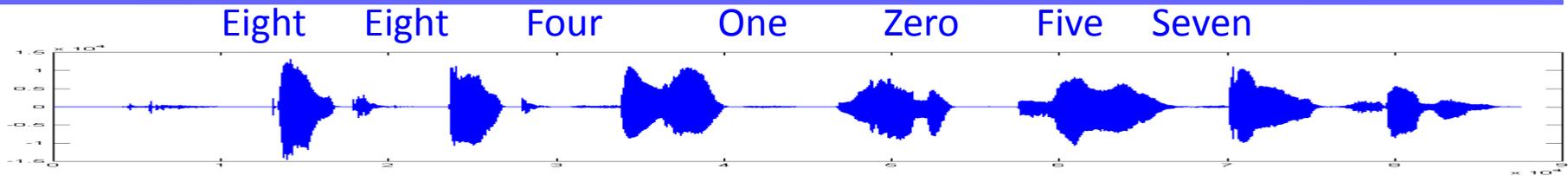
**Trainer**  
 Learns characteristics  
 of sound units

**Decoder**  
 Identifies sound units based  
 on learned characteristics

Map words into phoneme  
 sequences  
 and learn models for  
 phonemes  
 New words can be added  
 to the dictionary  
**AND RECOGNIZED**



# Words vs. Phonemes



**Unit = whole word**

**Average training examples per unit =  $7/6 \approx 1.17$**

**ey t ey t f ow r w a n z iy r ow f ay v s e v e n**

**Unit = sub-word**

**Average training examples per unit =  $22/14 \approx 1.57$**

More training examples = better statistical estimates of model (HMM) parameters

The difference between training instances/unit for phonemes and words increases dramatically as the training data and vocabulary increase

# How do we define phonemes?

---

- The choice of phoneme set is not obvious
  - Many different variants even for English
- Phonemes should be different from one another, otherwise training data can get **diluted**
  - Consider the following (hypothetical) example:
    - Two phonemes “AX” and “AH” that sound nearly the same
      - If during training we observed 5 instances of “AX” and 5 of “AH”
      - There might be insufficient data to train either of them properly
      - However, if both sounds were represented by a common symbol “A”, we would have 10 training instances!

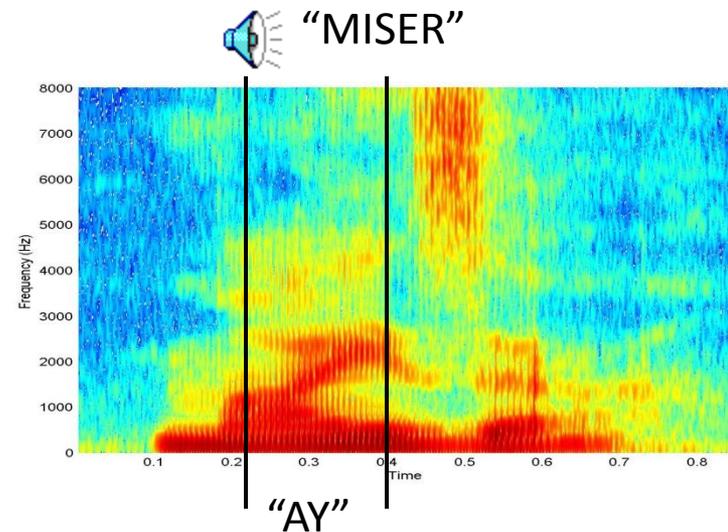
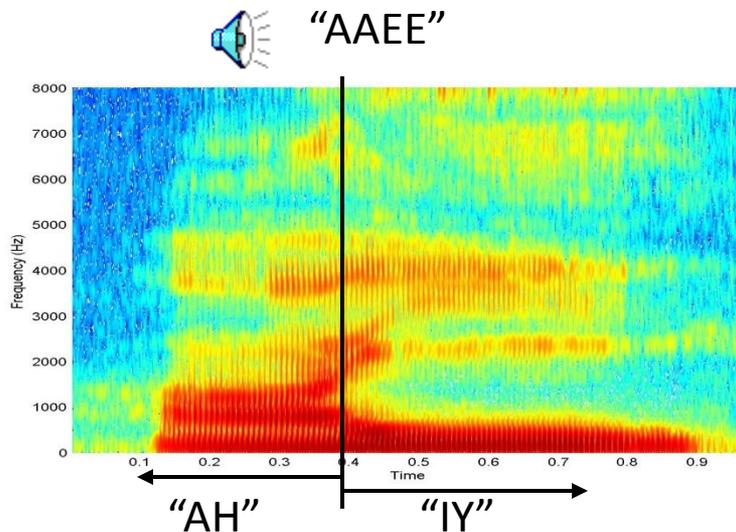
# Defining Phonemes

---

- They should be **significantly different** from one another to avoid inconsistent labelling
  - E.g. “AX” and “AH” are similar but not identical
  - ONE: W AH N
    - AH is clearly spoken
  - BUTTER: B AH T AX R
    - The AH in BUTTER is sometimes spoken as AH (clearly enunciated), and at other times it is very short “B AH T AX R”
    - The entire range of pronunciations from “AX” to “AH” may be observed
      - Not possible to make clear distinctions between instances of B AX T and B AH T
    - Training on many instances of BUTTER can result in AH models that are very close to that of AX!
      - Corrupting the model for ONE!

# Defining a Phoneme

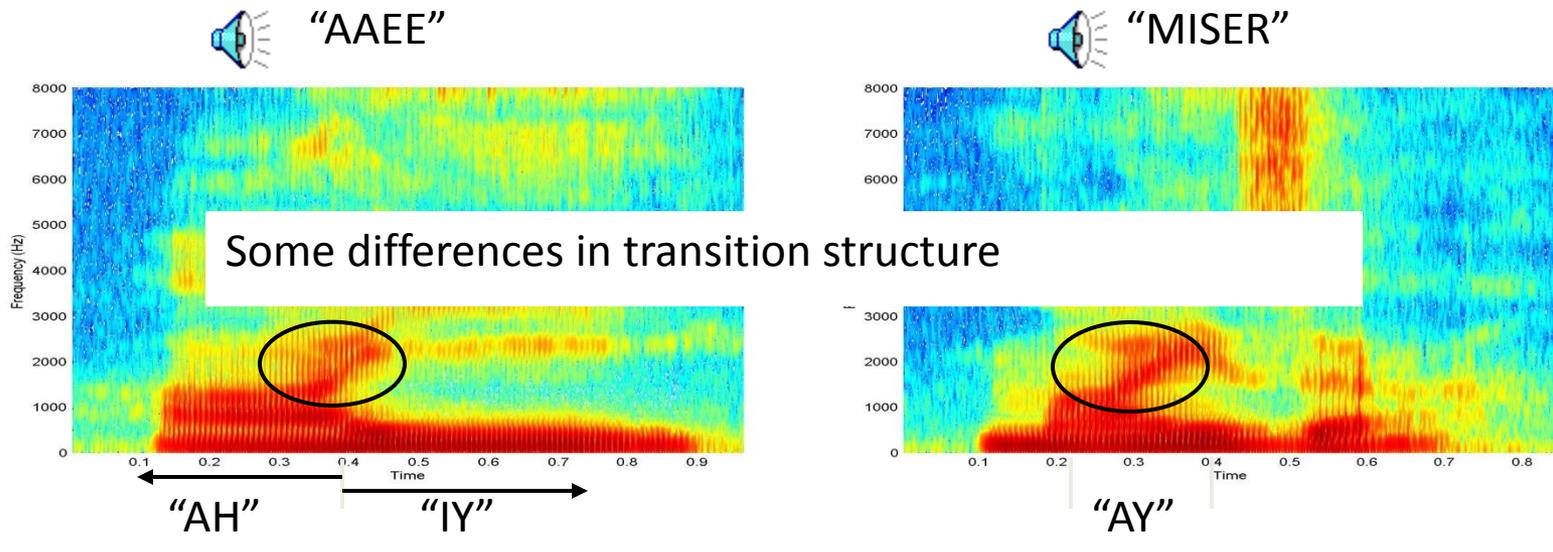
- Other inconsistencies are possible
  - Diphthongs are sounds that begin as one vowel and end as another, e.g. the sound “AY” in “MY”
  - Must diphthongs be treated as pairs of vowels or as a single unit?
  - An example



- Is the sound in Miser the sequence of sounds “AH IY”, or is it the diphthong “AY”

# Defining a Phoneme

- Other inconsistencies are possible
  - Diphthongs are sounds that begin as one vowel and end as another, e.g. the sound “AY” in “MY”
  - Must diphthongs be treated as p of vowels or as a single unit?
  - An example



- Is the sound in Miser the sequence of sounds “AH IY”, or is it the diphthong “AY”

# A Rule of Thumb

---

- If compound sounds occur frequently and have smooth transitions from one phoneme to the other, the compound sound can be single sound
  - Diphthongs have a smooth transition from one phoneme to the next
    - Some languages like Spanish have no diphthongs – they are always sequences of phonemes occurring across syllable boundaries with no guaranteed smooth transitions between the two
- Diphthongs: AI, EY, OY (English), UA (French) etc.
  - Different languages have different sets of diphthongs
- Stop sounds have multiple components that go together
  - A closure, followed by burst, followed by frication (in most cases)
- Some languages have *triphthongs*

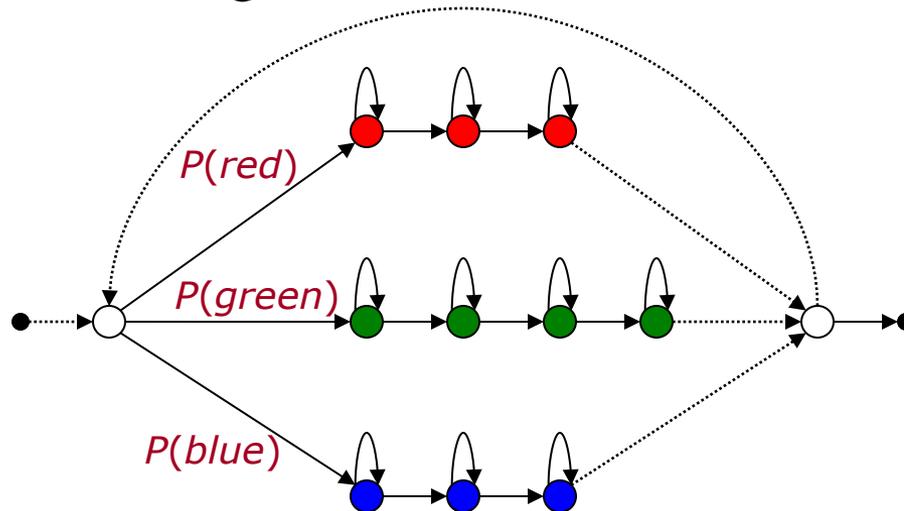
# Phoneme Sets

---

- Conventional Phoneme Set for English:
  - Vowels: AH, AX, AO, IH, IY, UH, UW etc.
  - Diphthongs: AI, EY, AW, OY, UA etc.
  - Nasals: N, M, NG
  - Stops: K, G, T, D, TH, DH, P, B
  - Fricatives and Affricates: F, HH, CH, JH, S, Z, ZH etc.
- Different groups tend to use a different set of phonemes
  - Varying in sizes between 39 and 50!

# Recognition with Subword Units

- Word based recognition Review:



- Create a large “grammar” HMM using the HMMs for individual words
- Find the best state sequence through the grammar HMM
- This also gives us the best word sequence automatically

# Recognition with Phonemes

---

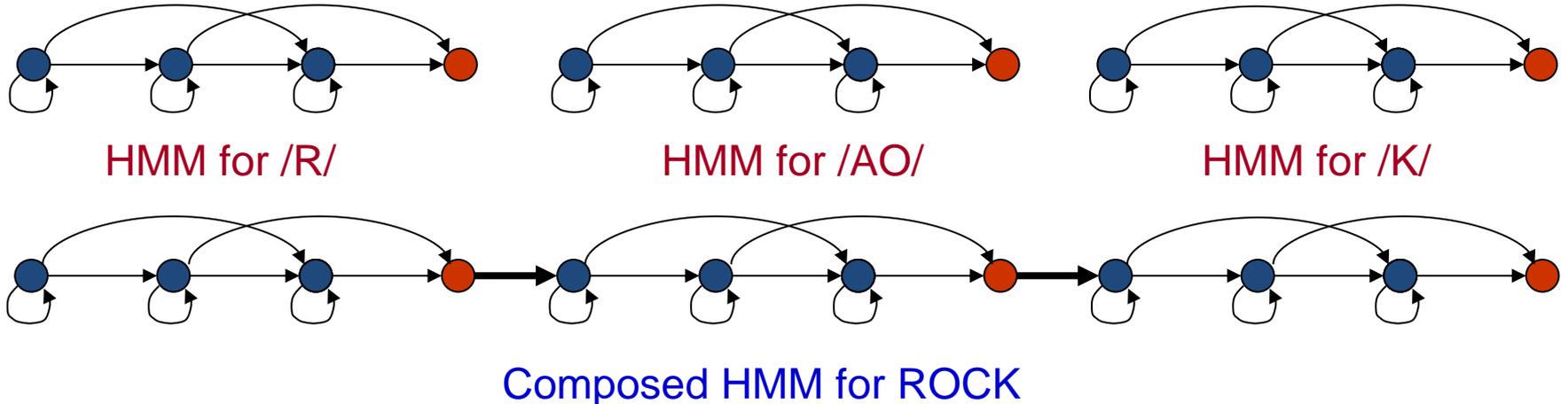
- The phonemes are only meant to enable better *learning* of templates
  - HMM or DTW models
- **We still recognize *words***
- The models for words are composed from the models for the subword units
- The HMMs for individual words are connected to form the Grammar HMM
- The best word sequence is found by Viterbi decoding

# Recognition with phonemes

*Example:*

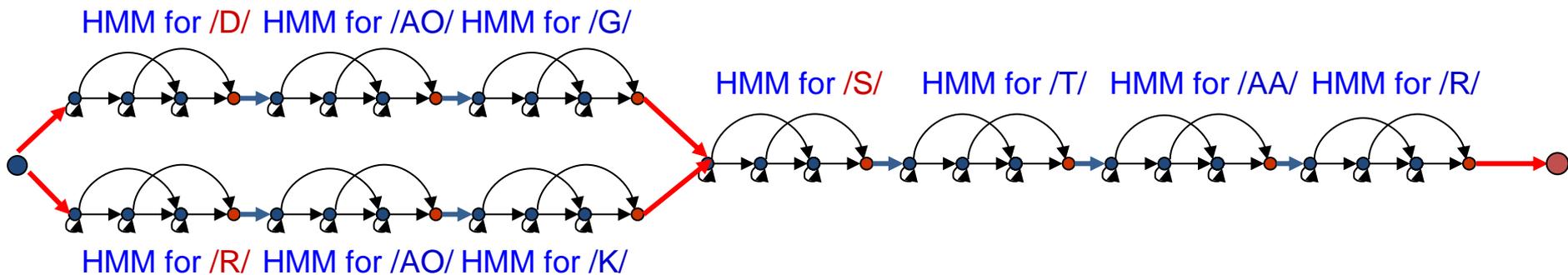
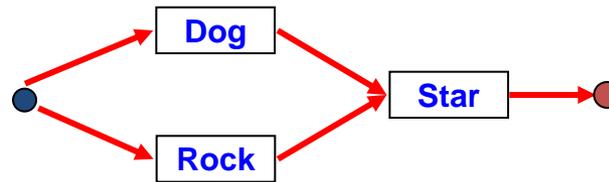
<u>Word</u>	<u>Phones</u>
Rock	R AO K

- Each phoneme is modeled by an HMM
- Word HMMs are constructed by concatenating HMMs of phonemes
- Composing word HMMs with phoneme units does not increase the complexity the grammar/language HMM



# Recognition with phonemes

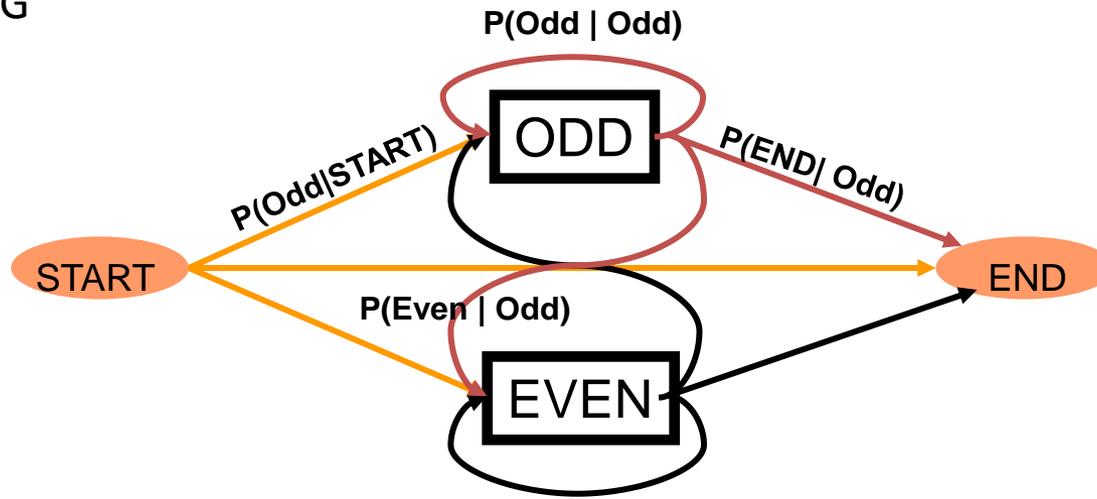
A simple grammar that recognizes either the phrase  
“DOG STAR” or “ROCK STAR”



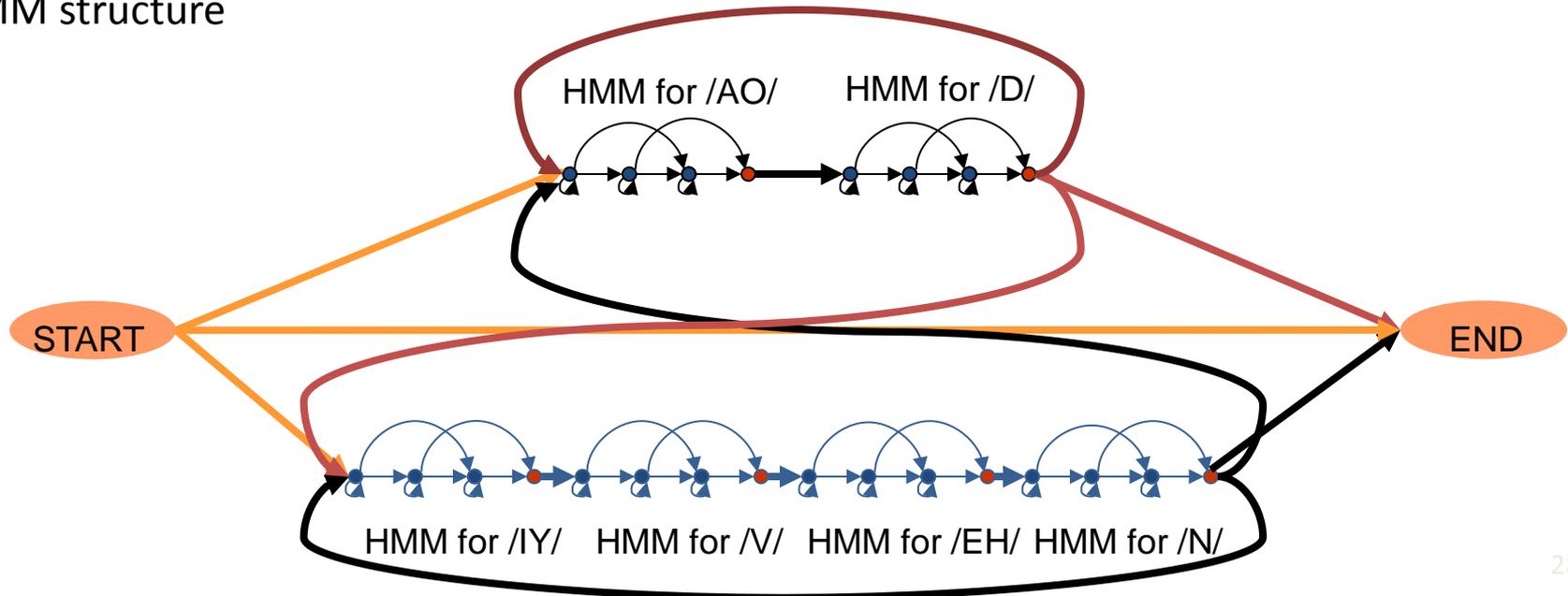
Thick lines represent connections between phonemes  
and connections between words

# Recognizing Odd vs. Even

This word-level FSG



Translates to this HMM structure



# Building Word Model Networks

---

- Procedure for building word model networks:
  1. Compose word graph from grammar
  2. For each word in the graph, derive the pronunciation from a pronunciation dictionary
  3. Compose HMM for the word from the HMMs for the phonemes
  4. Replace each edge representing a word in the graph by the HMM for that word
  5. Decode

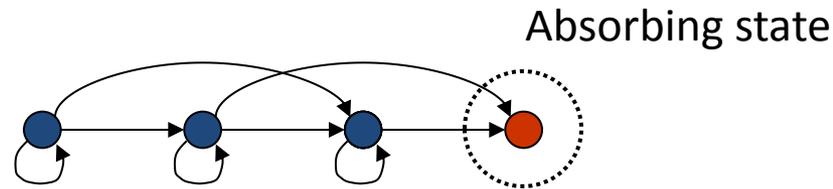
# Composing a Word HMM

---

- Words are linear sequences of phonemes
- To form the HMM for a word, the HMMs for the phonemes must be linked into a larger HMM
- Two mechanisms:
  - Explicitly maintain a *non-emitting* state between the HMMs for the phonemes
    - Computationally efficient, but complicates time-synchronous search
  - Expand the links out to form a sequence of *emitting-only* states

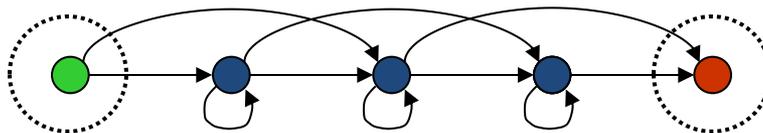
# Linking Phonemes via Non-emitting State

- For every phoneme any state sequence can terminate in any state  $s$  with a probability  $P_s$  (phoneme) (which can be 0).
  - This is often represented as a transition into an absorbing state with probability  $P_s$  (phoneme)



- For every phoneme any state sequence can begin at any state  $s$  with a probability  $\pi_s$  (phoneme) (which can be 0)
  - This can be represented as transitions from a generating state with probability  $\pi_s$  (phoneme)

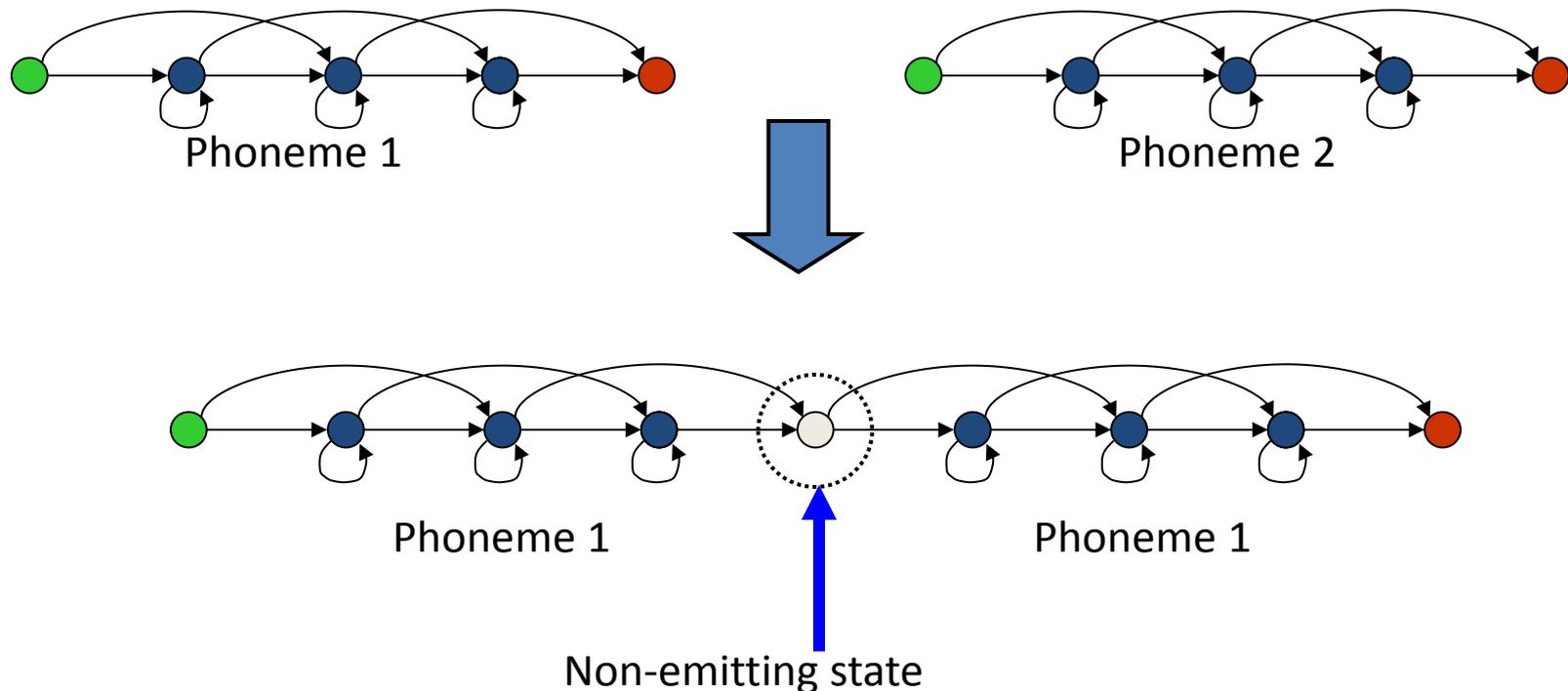
Generating state



- Often we only permit the first state of the HMM to have non-zero  $\pi_s$ . In this case the generating state may be omitted

# Linking Phonemes via Non-emitting State

- To link two phonemes, we create a new “non-emitting” state that represents both the absorbing state of the first phoneme and the generating state of the second phoneme



# Linking Phonemes Directly

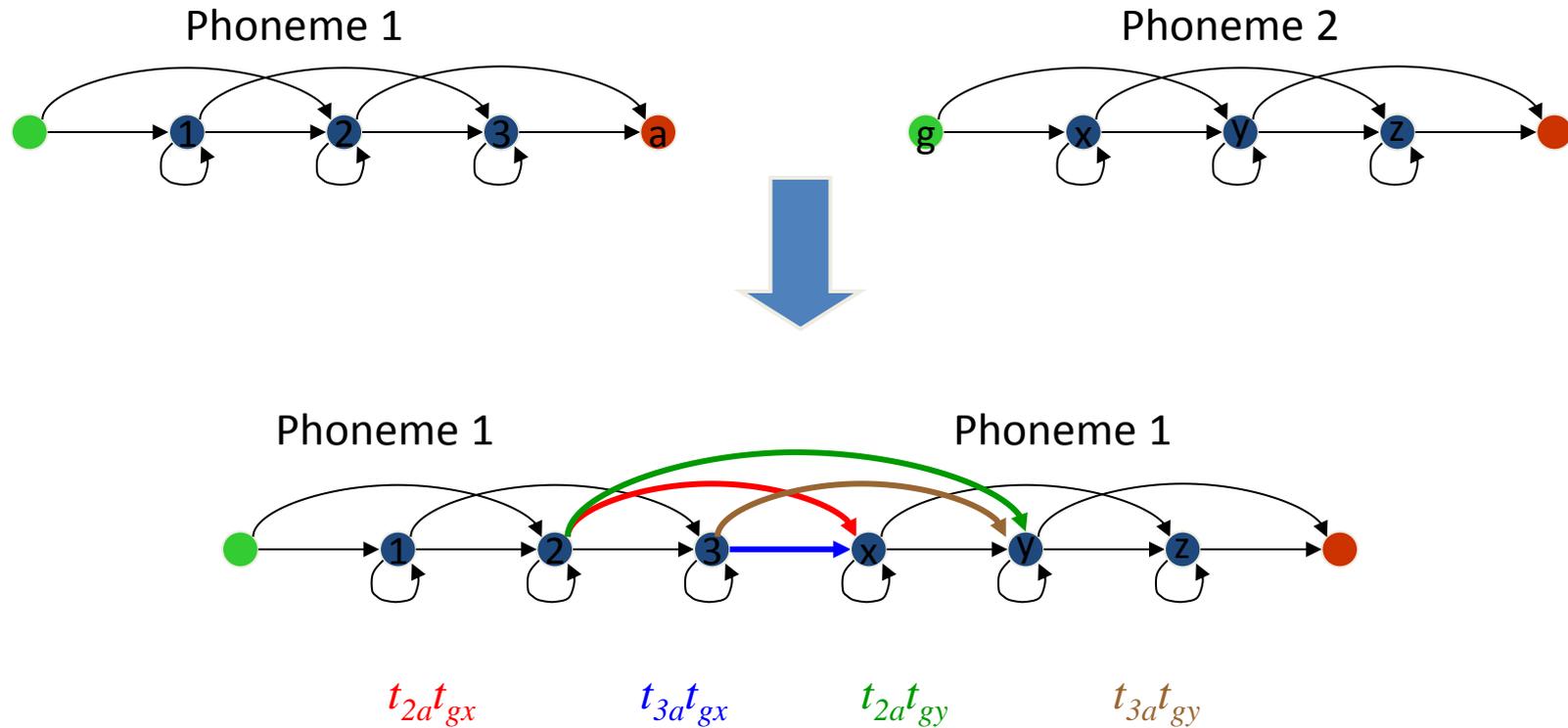
---

- Transitions into absorbing states of the first phoneme are factored into the transitions from the generating state of the second
- Let  $t_{ph1}(s,A)$  be transition probabilities from any state  $s$  to the absorbing state of phoneme 1
- Let  $t_{ph2}(G,s)$  be transition probabilities from the generating state of phoneme 2 into any state  $s$
- The joint HMM will now have direct transitions from state  $s_i$  of phoneme 1 to state  $s_j$  of phoneme 2 with probabilities

$$t_{ph1,ph2}(s_i, s_j) = t_{ph1}(s_i,A) t_{ph2}(G,s_j)$$

- Every combination of  $s_i$  and  $s_j$  must be considered when making this connection

# Linking Phonemes Directly



- Colour codes to show the probabilities of each transition

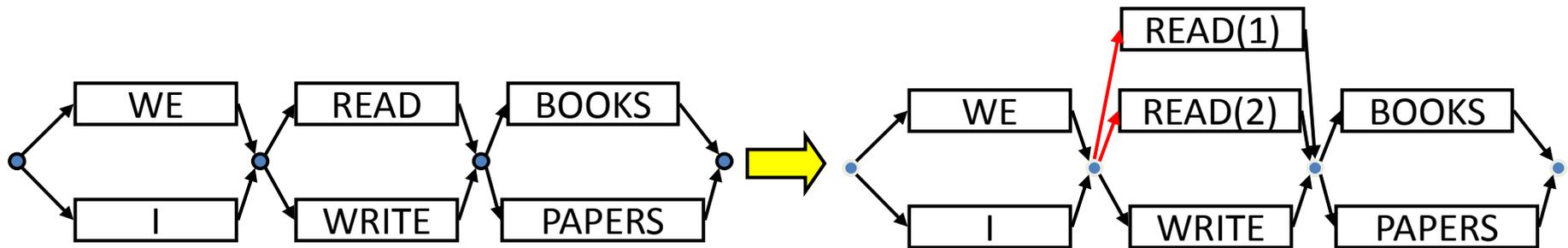
# The problem of pronunciation

---

- There are often multiple ways of pronouncing a word.
  - Sometimes these pronunciation differences are semantically meaningful:
    - READ : R IY D (Did you read the book)
    - READ : R EH D (Yes I read the book)
  - At other times they are not
    - AN : AX N (That's an apple)
    - AN : AE N (An apple)
- These are typically identified in a dictionary through markers
  - READ(1) : R IY D
  - READ(2) : R EH D

# Handling multiple pronunciations

- While building a “grammar” HMM, every pronunciation of the word must be separately considered
  - Each instance of a word in the grammar will be represented by multiple parallel paths in the finite state graph, one for each pronunciation
  - E.g. (I | WE) (READ | WRITE) (BOOKS | PAPERS)



- If edges have probabilities, every copy of the word will carry the same probability

# Handling multiple pronunciations

---

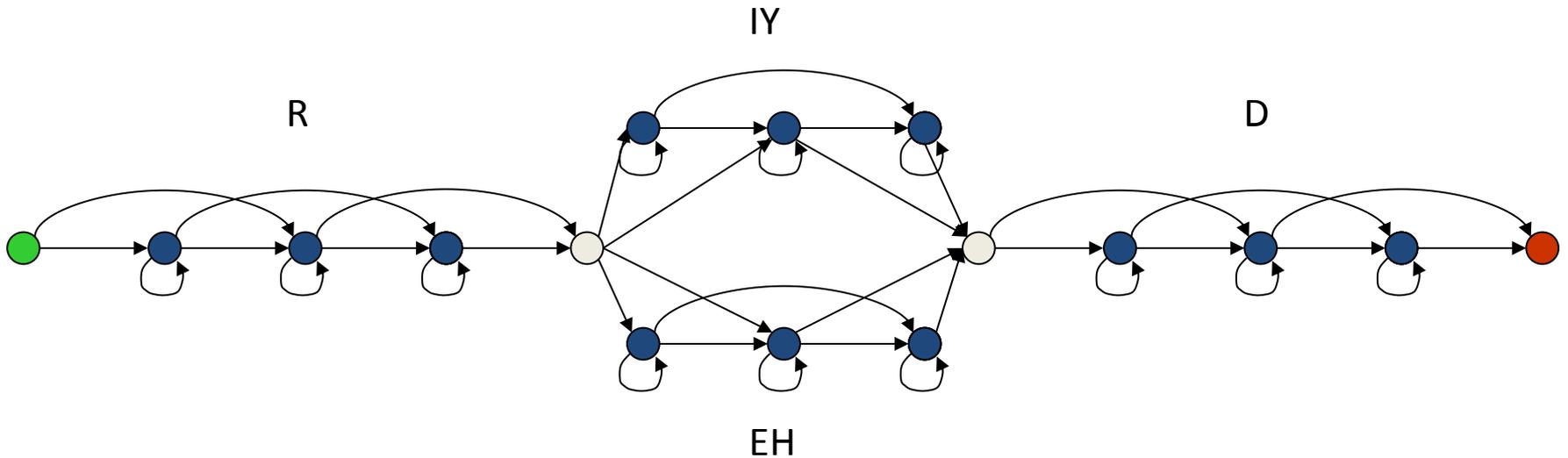
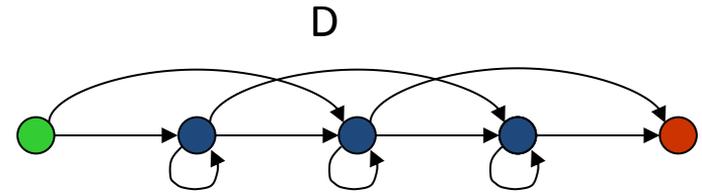
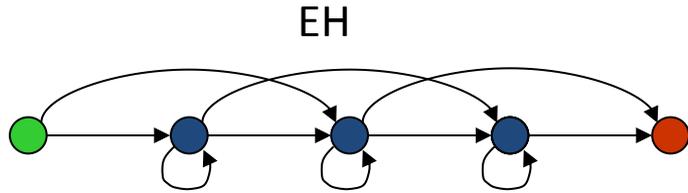
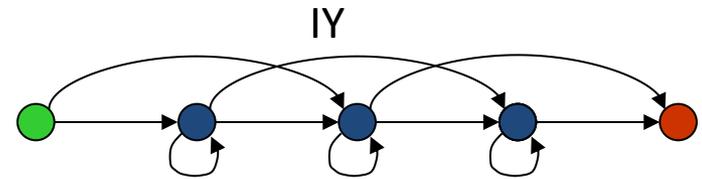
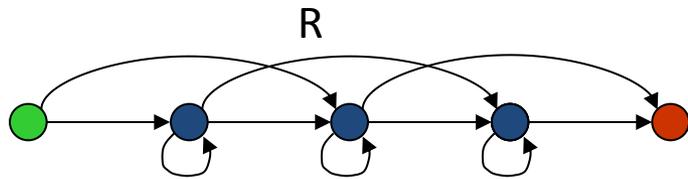
- Multiple word pronunciations may be more compactly represented by graphs
  - READ : R (EH | IY) D
  - AN : (AX | AE) N
- In this case there is no need to create multiple parallel edges in the grammar to represent multiple pronunciations
- However, the HMM for the word itself becomes more complex

# HMM for word with multiple prons.

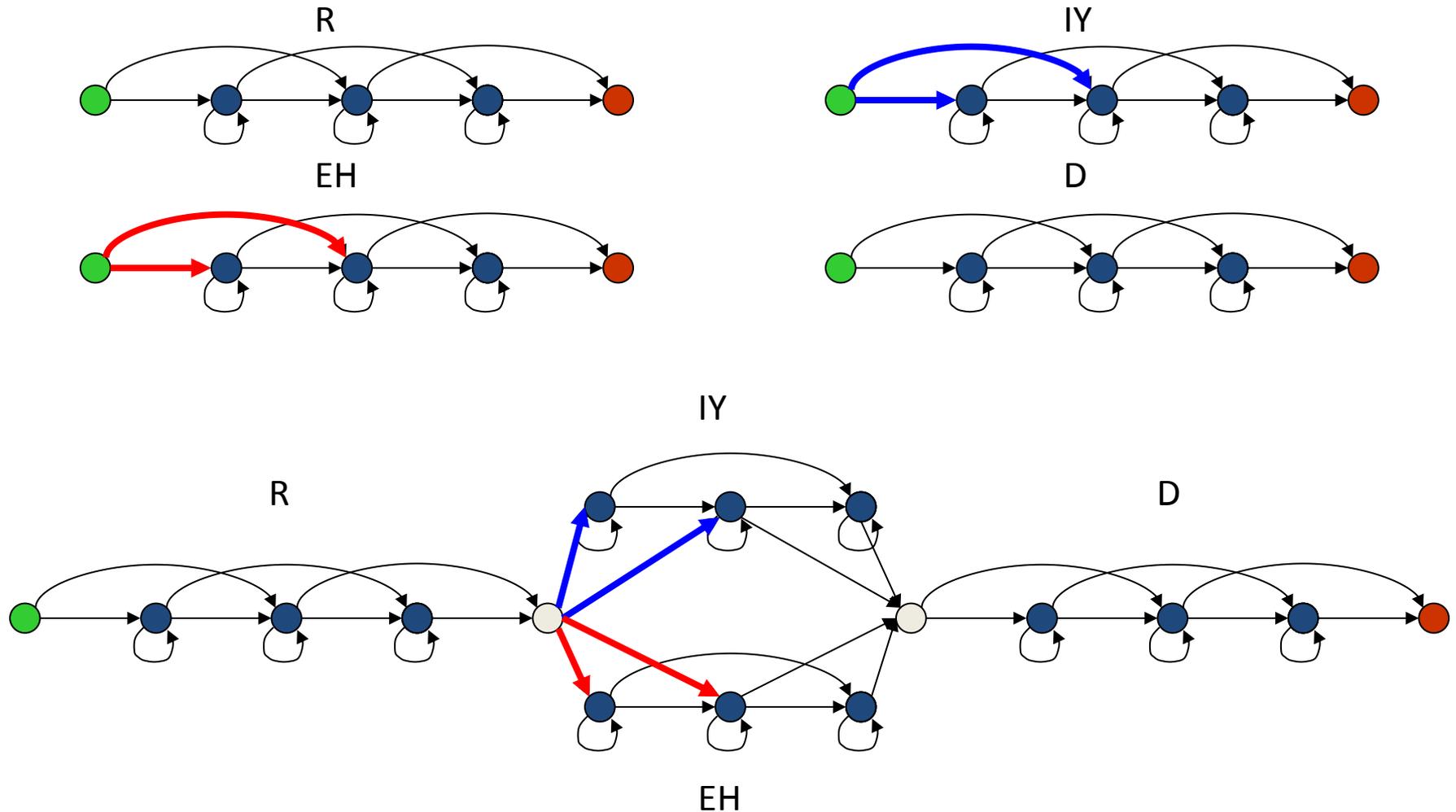
---

- The HMM for the word must represent the pronunciation graph
  - HMM construction must consider branching into / merging from multiple phonemes
- Once again, the HMM could be constructed either via non-emitting states or by direct linkage
- The HMM for READ : R (EH | IY ) D
  - HMM FOR R, HMM for EH, HMM for IY, HMM for D
  - HMM for READ
  - Using non-emitting states

# “READ” using non-emitting states

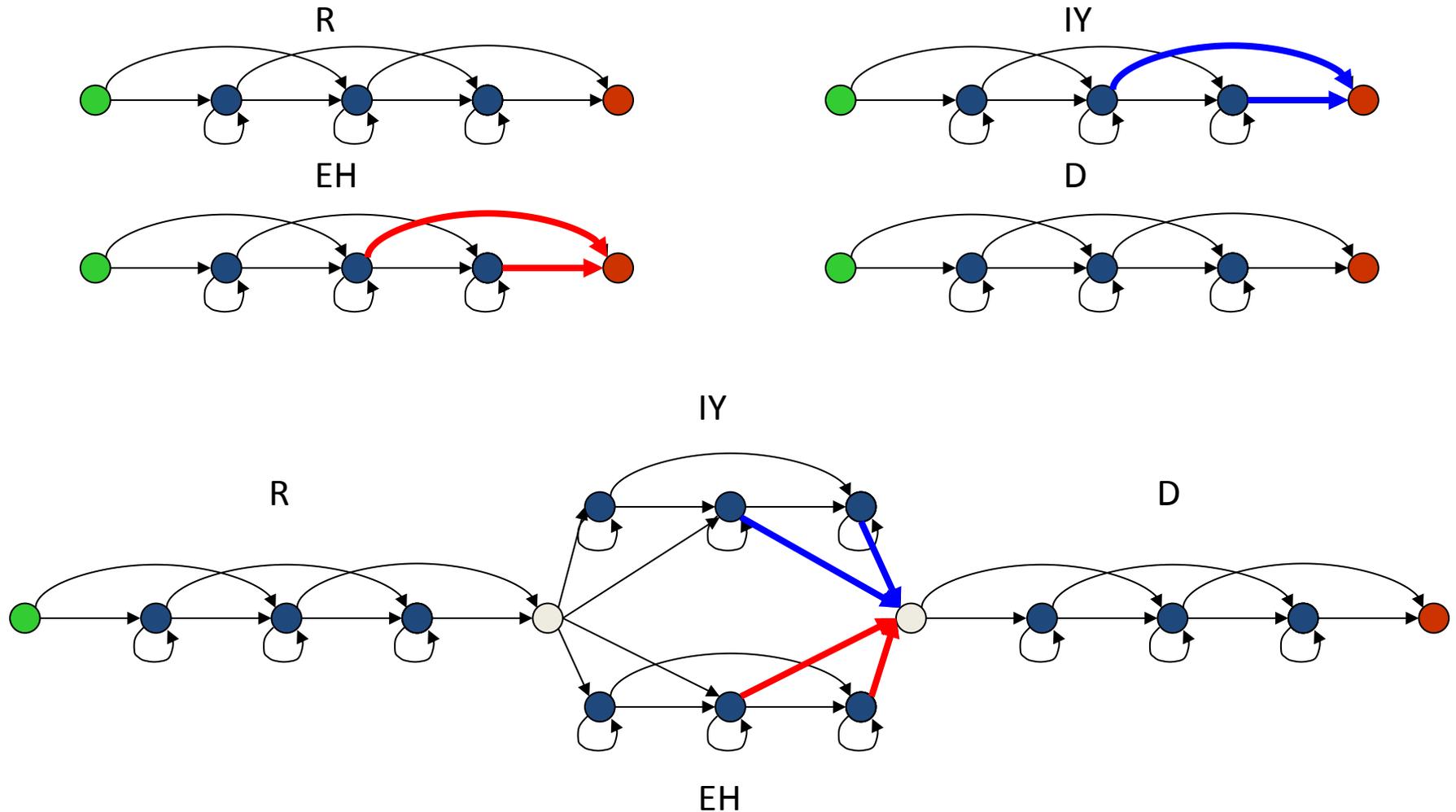


# “READ” using non-emitting states



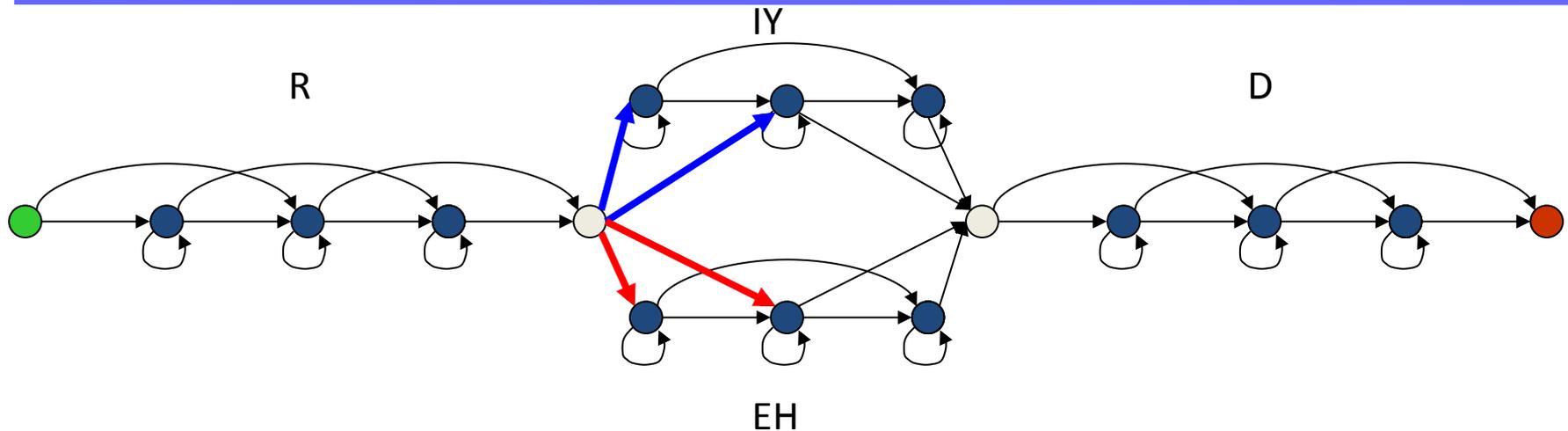
- Transition probabilities *from* non-emitting state into EH and IY are identical to transition probabilities from the original generating states of EH and IY

# “READ” using non-emitting states



- Transition probabilities *into* non-emitting state from EH and IY are identical to transition probabilities into the original absorbing states of EH and IY

# “READ” using non-emitting states



- A break from theory:
  - The sum of all outgoing transition probabilities from any state should theoretically sum to 1.0
  - Here, however, the probabilities of the two blue arcs sum to 1.0, as do the probabilities of the two red arcs
  - The total probability of all outgoing arcs from the first non-emitting state is greater than 1.0
- The probabilities may be normalized to sum to 1.0 for theoretical consistency, but practically, this is often not effective

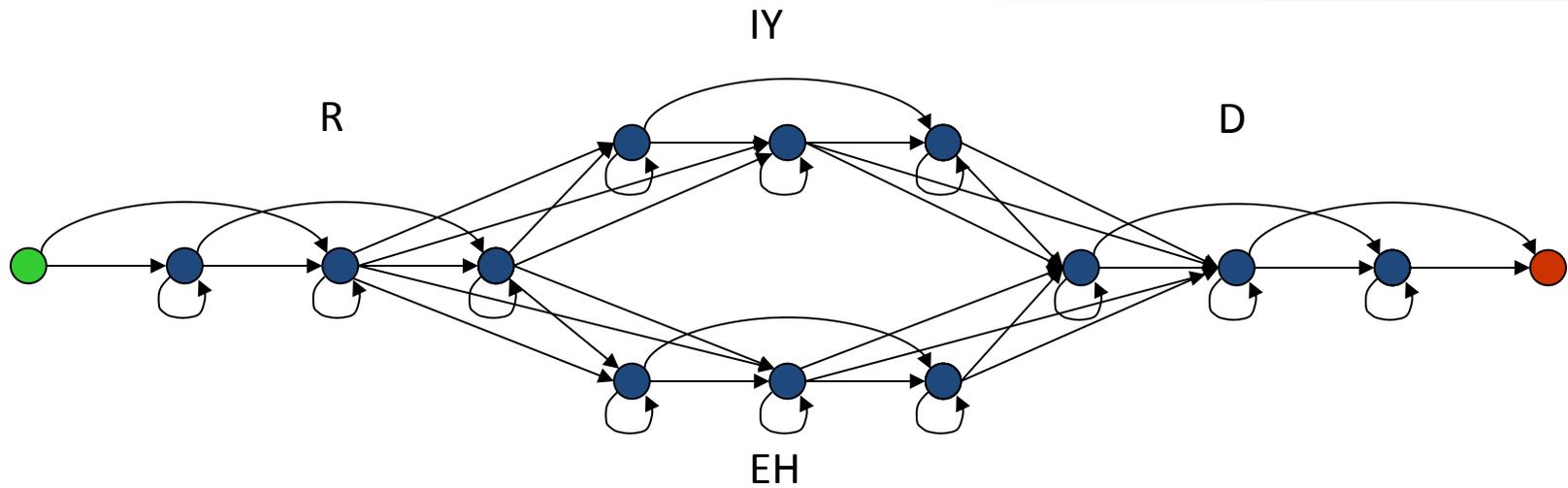
# Building with direct connections

---

- Phonemes may be linked via direct connection
  - No non-emitting states
- The HMM structure now becomes more complex
  - All transitions into multiple phonemes ( $R \rightarrow [IY, EH]$ ) at branching points must be considered
  - All transitions from multiple phonemes ( $[IY, EH] \rightarrow D$ ) must be considered

# Fully connected HMM for READ

---



- Save your time ..

# It can get more complex

---

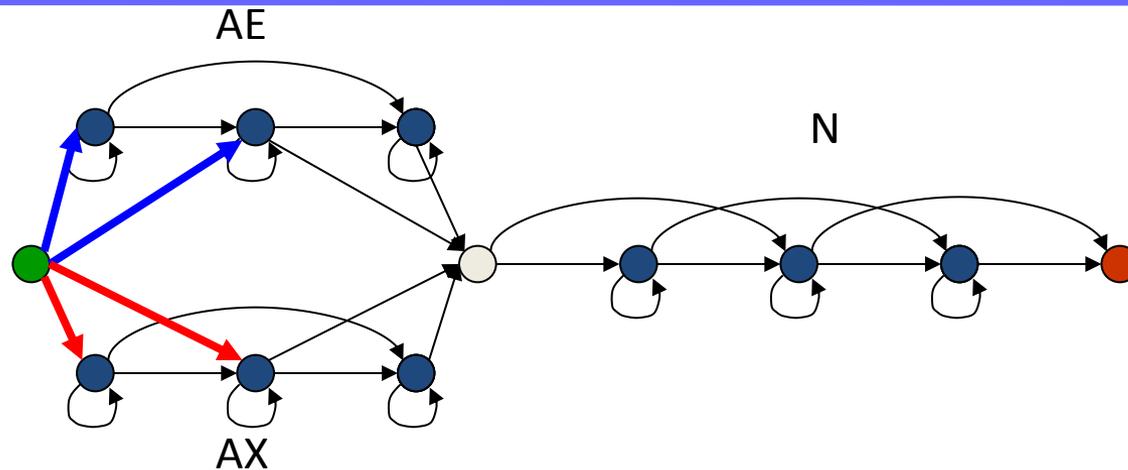
- ALLENBY : AE L (EH | AX) (N | M) B IY
- This is best done with non-emitting states
- Directly linking the phonemes without non-emitting states can result in the addition of a very large number of transitions

# Multiple phones at the begin/end

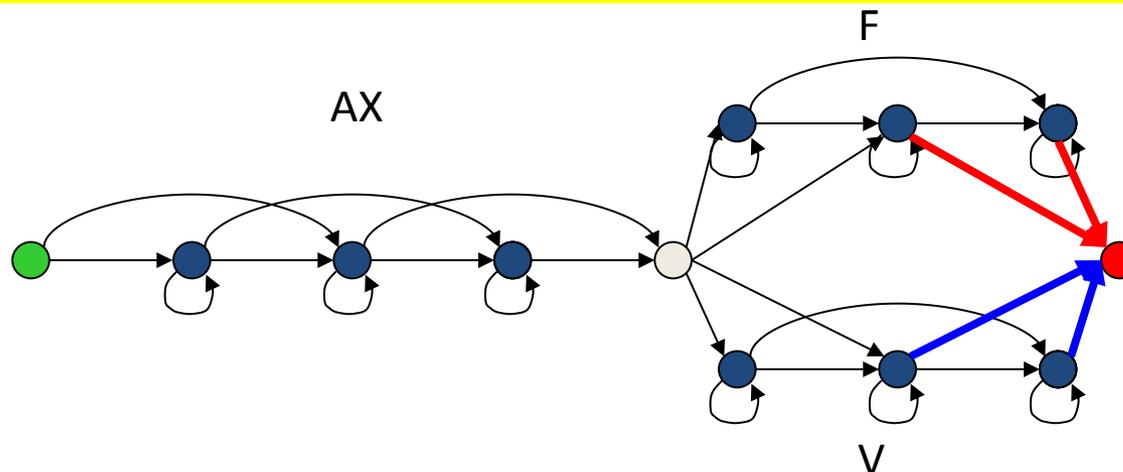
---

- Multiple alternative phonemes may be likely at the beginnings and ends of words
  - (AX | AE) N
- Multiple phonemes at the beginning: The number of states in the HMM with non-zero initial probabilities will be the sum of the number of such states in each of the entry phonemes
  - I.e. no. of non-zero initial prob. states in AX + the no. of non-zero initial prob. states in AE
  - These can be represented as transitions out of a generating state for the word

# Multiple phones at the begin/end

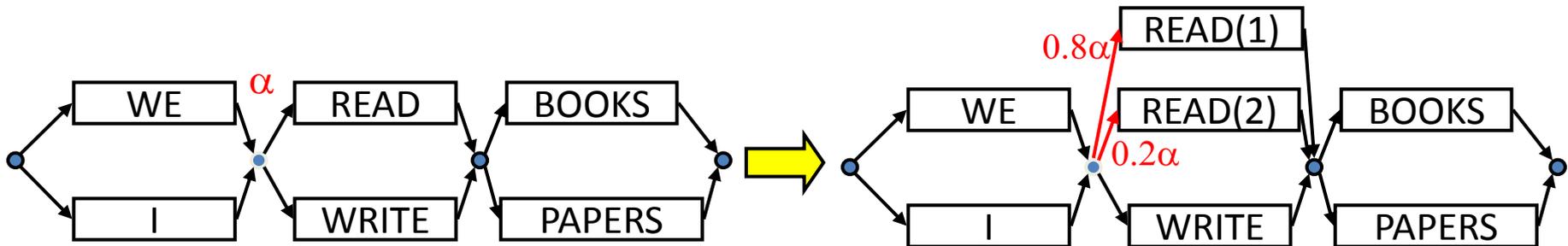


- Similarly, the probability of terminating in the various states of pronunciations with multiple alternate terminal phonemes (e.g. OF: AX (F | V)) can be represented through a common absorbing state



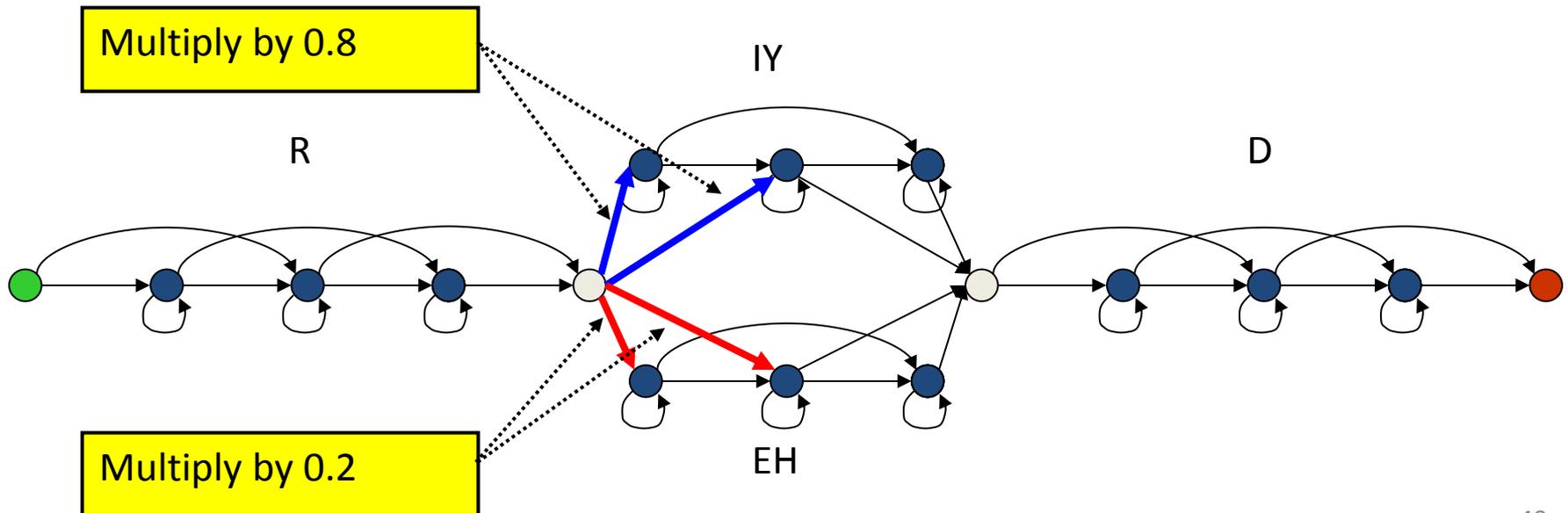
# Associating Probabilities With Prons

- Some pronunciations are more probable than others. We may want to incorporate these a priori probabilities into the search
  - E.g. “AE N” is used much less frequently than “AX N”
  - $\text{Prob}(\text{AE N}) = 0.2$ ;  $\text{Prob}(\text{AX N}) = 0.8$
- If multiple pronunciations are represented by multiple edges, then we simply associate an additional probability with each edge



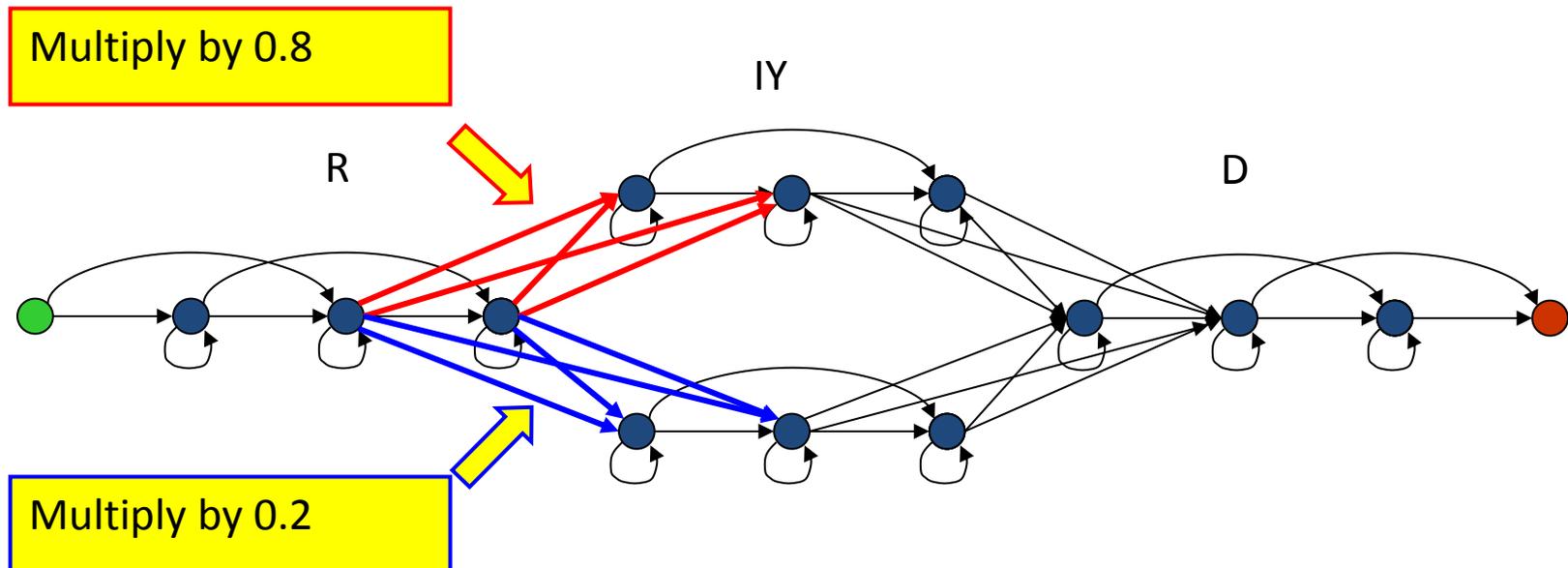
# Associating Probabilities With Prons

- For phoneme networks, probabilities will be part of the graph:
  - READ: R (IY <0.8> | EH <0.2>) D
- The probabilities can be factored into the transitions between states of phonemes



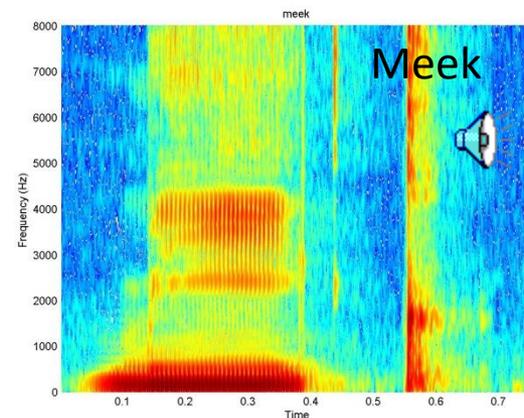
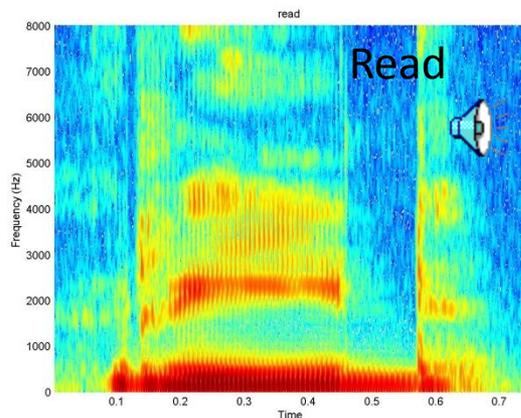
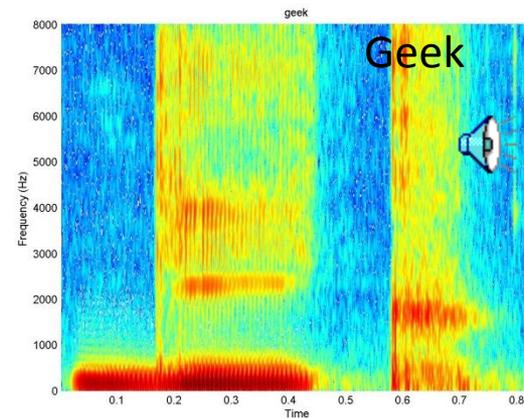
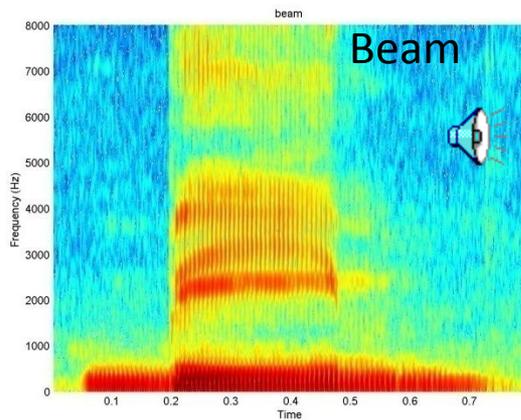
# Associating Probabilities With Prons

- For phoneme networks, probabilities will be part of the graph:
  - READ: R (IY <0.8> | EH <0.2>) D
- The probabilities can be factored into the transitions between states of phonemes



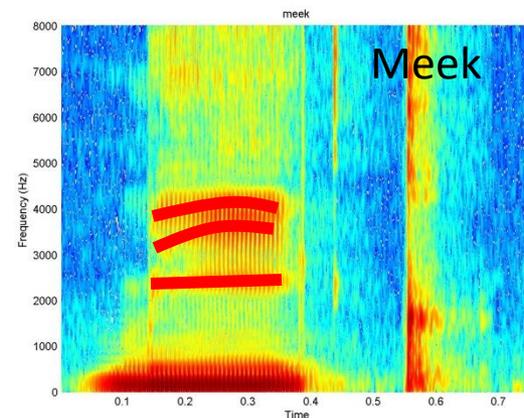
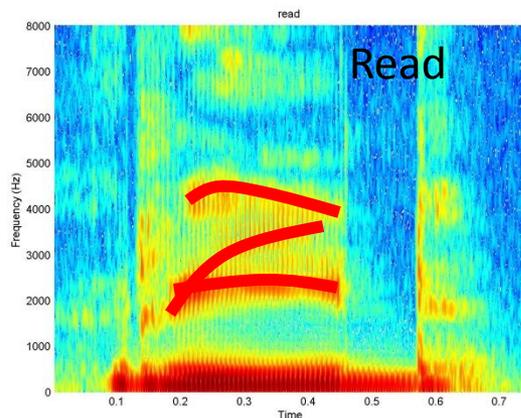
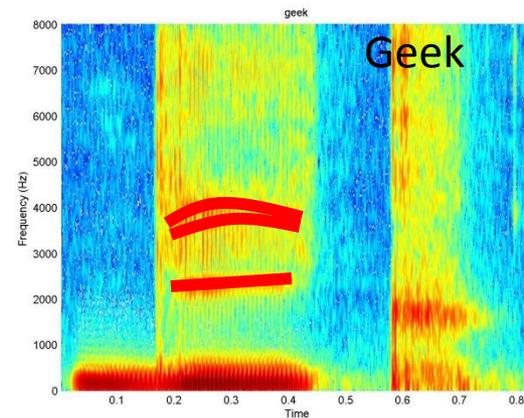
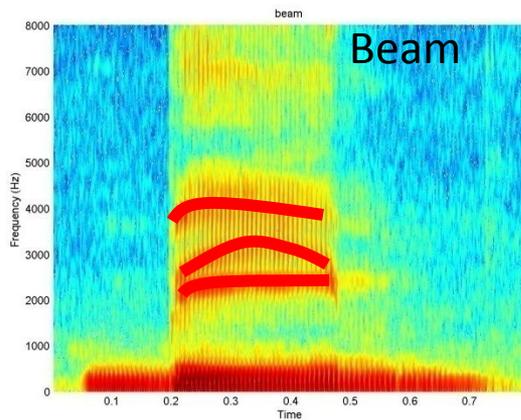
# The Effect of Context

- Phonemes are not entirely consistent
  - Different instances of a phoneme will differ according to its neighbours
  - E.g: Spectrograms of /y/ in different contexts



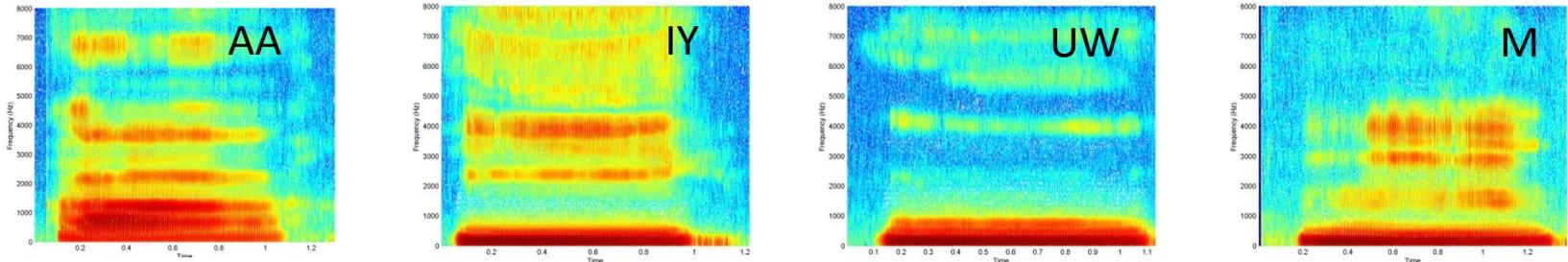
# The Effect of Context

- Phonemes are not entirely consistent
  - Different instances of a phoneme will differ according to its neighbours
  - E.g: Spectrograms of /ɪ/ in different contexts

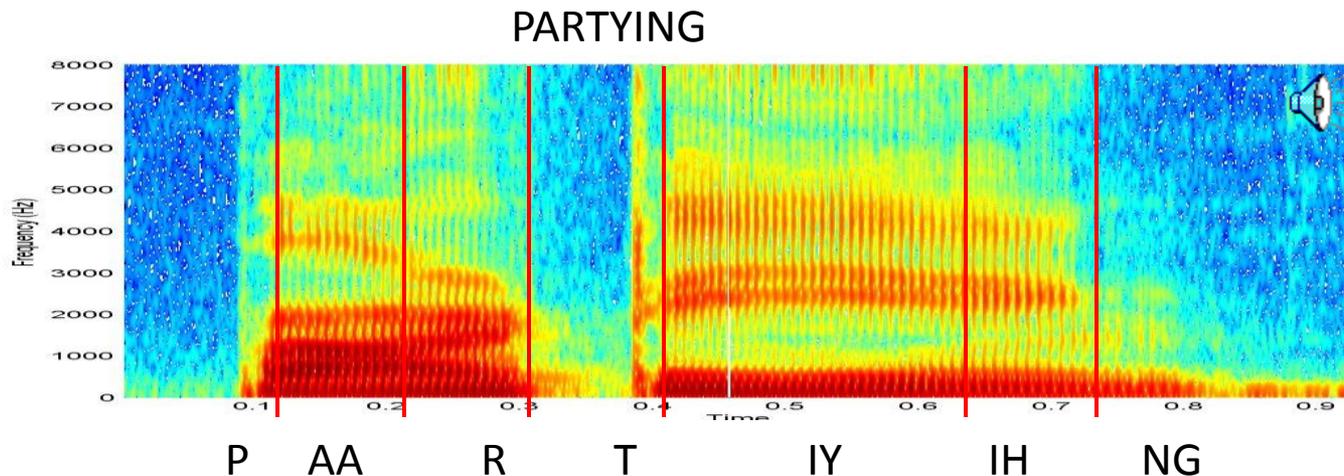


# The Effect of Context

- Every phoneme has a locus
  - The spectral shape that would be observed if the phoneme were uttered in isolation, for a long time

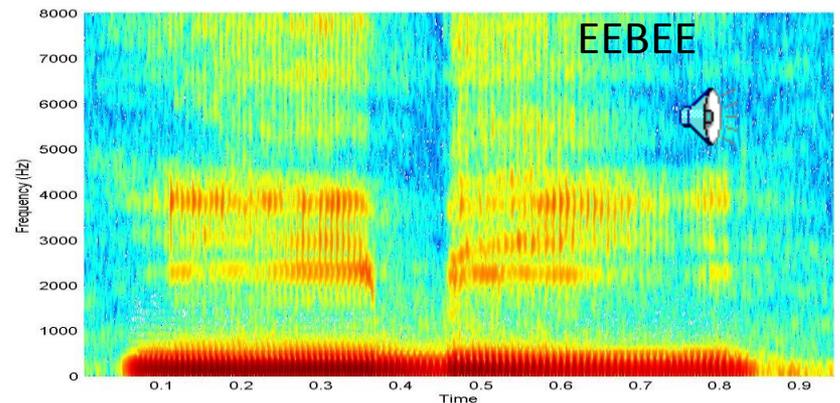
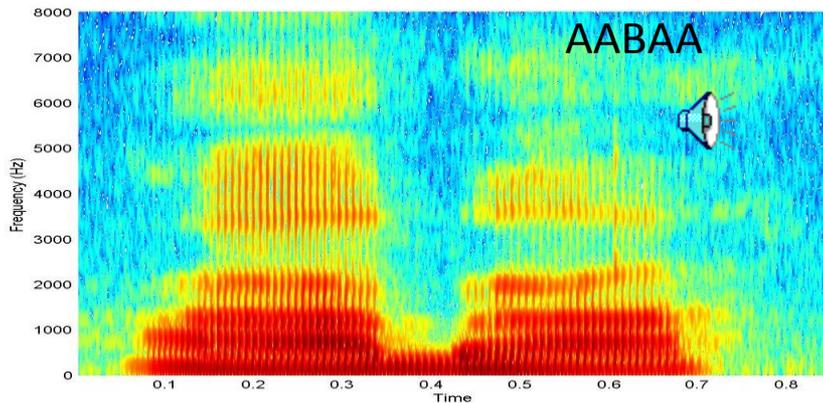


- In continuous speech, the spectrum attempts to arrive at locus of the current sound



# The Effect of Context

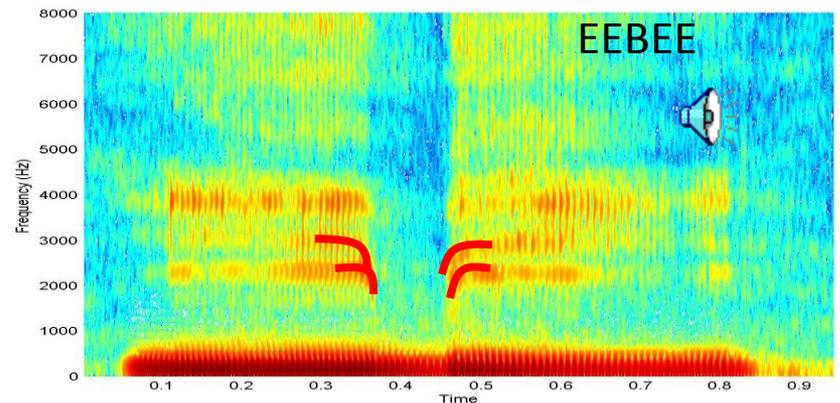
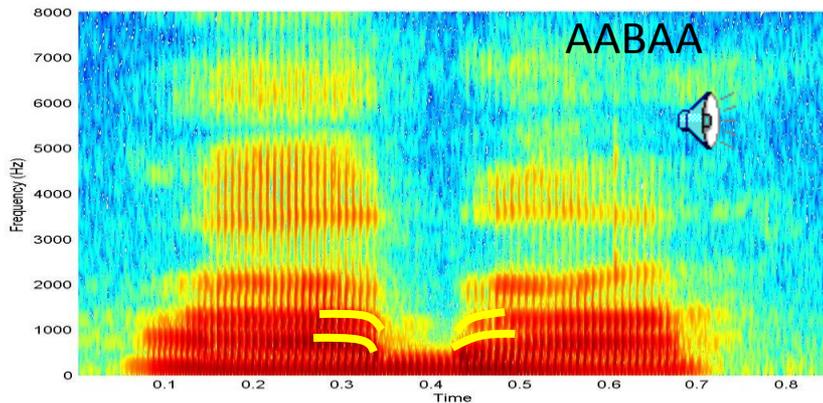
- Every phoneme has a locus
  - For many phoneme such as “B”, the locus is simply a virtual target that is never reached



- o Nevertheless, during continuous speech, the spectrum for the signal tends towards this virtual locus

# The Effect of Context

- Every phoneme has a locus
  - For many phoneme such as “B”, the locus is simply a virtual target that is never reached



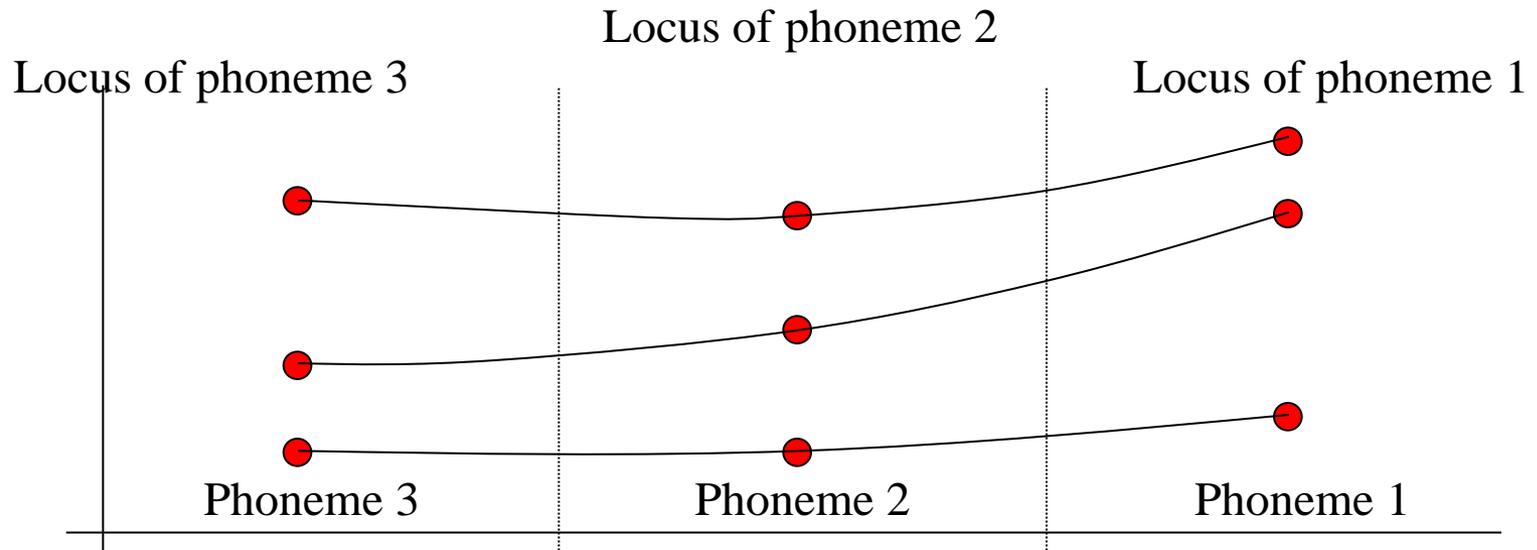
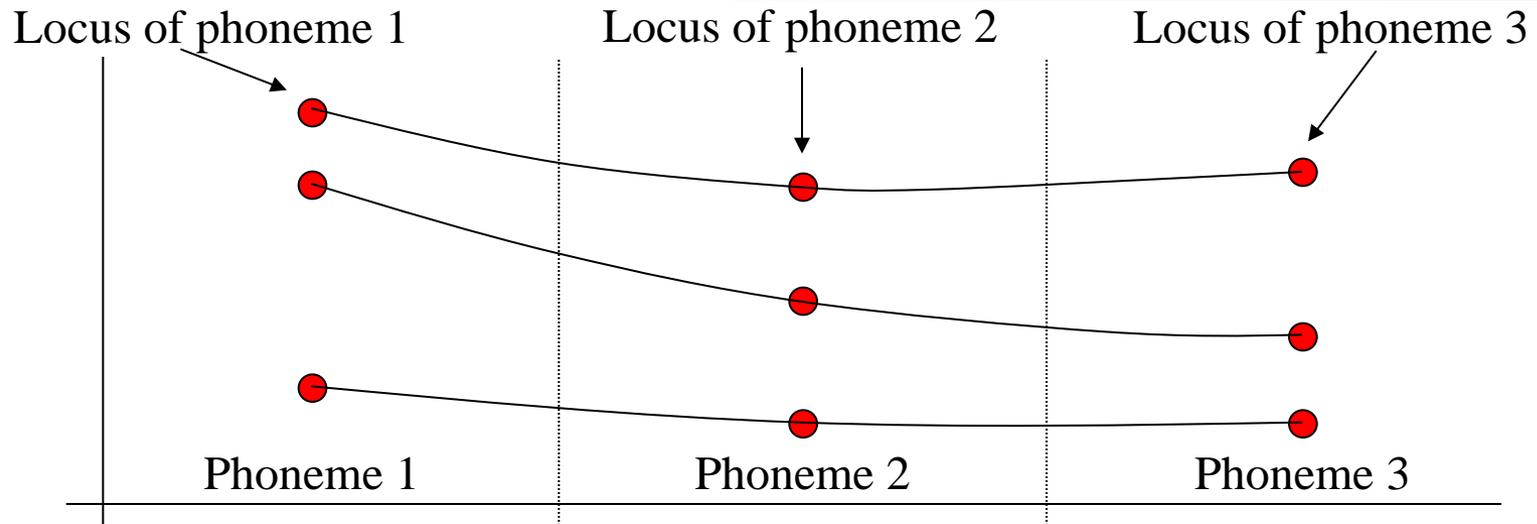
- o Nevertheless, during continuous speech, the spectrum for the signal tends towards this virtual locus

# Variability among Sub-word Units

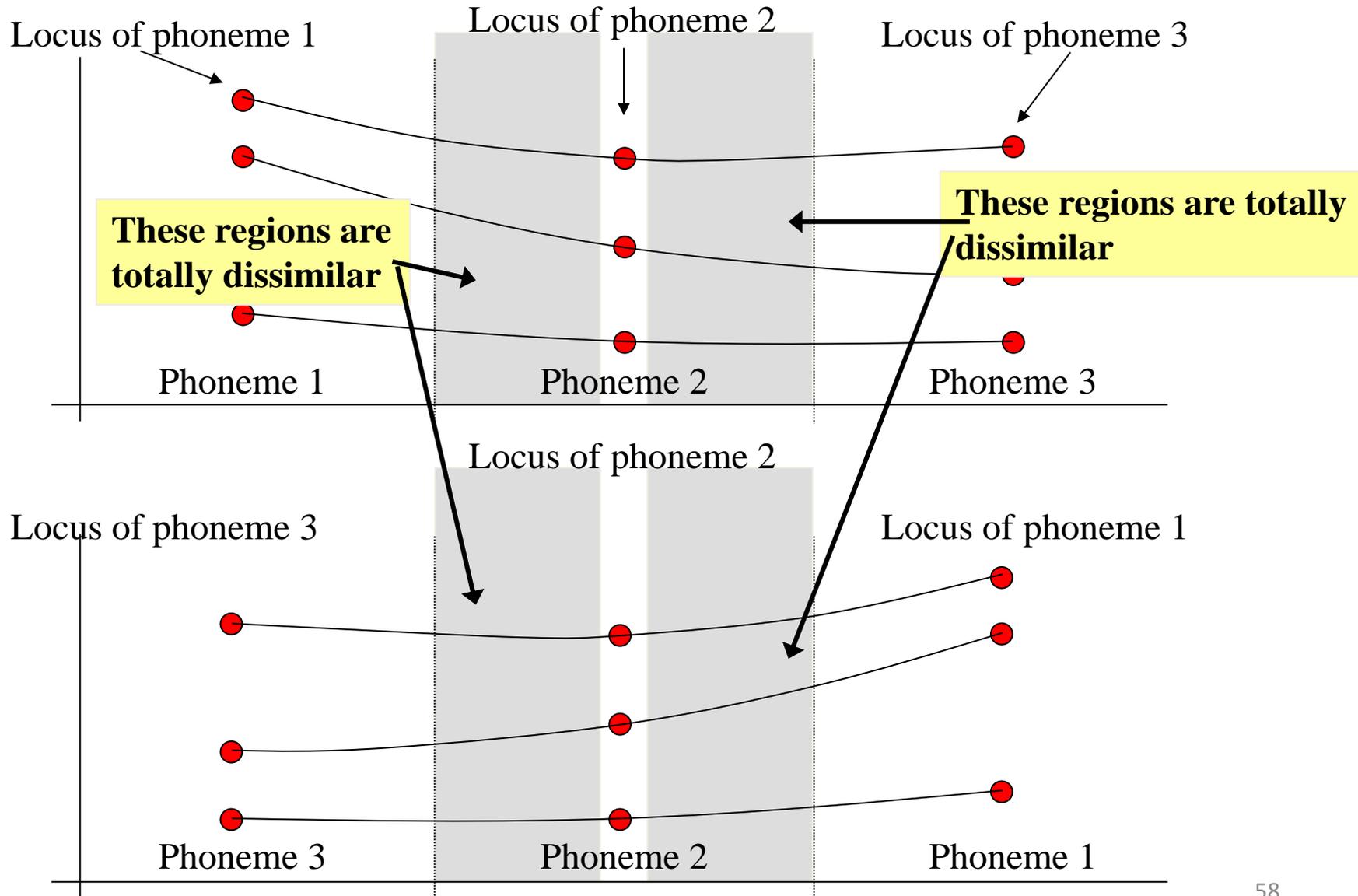
---

- The acoustic flow for phonemes: every phoneme is associated with a particular articulator configuration
- The spectral pattern produced when the articulators are exactly in that configuration is the locus for that phoneme
- As we produce sequences of sounds, the spectral patterns shift from the locus of one phoneme to the next
  - The spectral characteristics of the next phoneme affect the current phoneme
- The inertia of the articulators affects the manner in which sounds are produced
  - The vocal tract and articulators are still completing the previous sound, even as we attempt to generate the next one
- As a result of articulator inertia, the spectra of phonemes vary with the adjacent phonemes

# Spectral trajectory of a phoneme is dependent on context



# Spectral trajectory of a phoneme is dependent on context

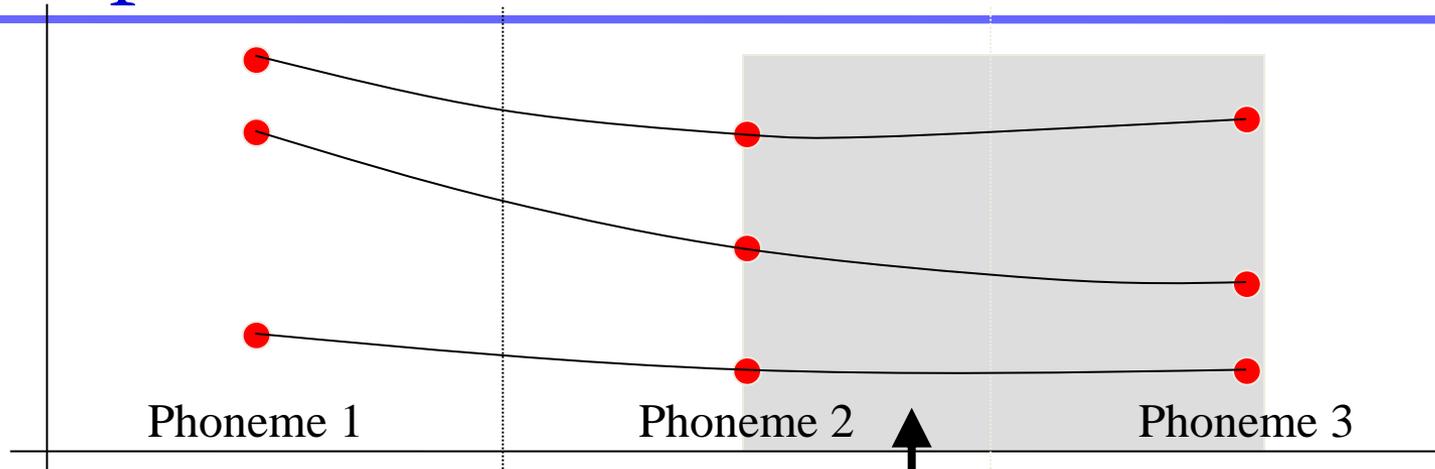


# Subword units with high variability are poor building blocks for words

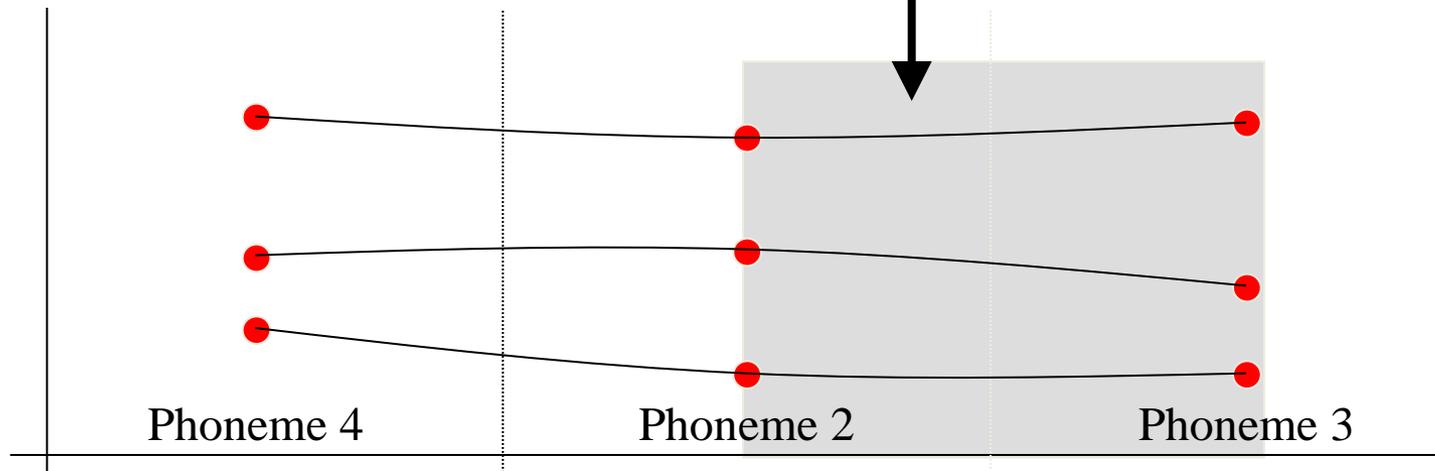
---

- Phonemes can vary greatly from instance to instance
- Due to co-articulation effects, some regions of phonemes are very similar to regions of other phonemes
  - E.g. the right boundary region of a phoneme is similar to the left boundary region of the next phoneme
- The boundary regions of phonemes are highly variable and confusable
  - They do not provide significant evidence towards the identity of the phoneme
  - Only the central regions, i.e. the loci of the phonemes, are consistent
- This makes phonemes confusable among themselves
  - In turn making these sub-word units poor building blocks for larger structures such as words and sentences
- Ideally, all regions of the sub-word units would be consistent

# Diphones – a different kind of unit



The shaded regions are similar, although the phonemes to the left are different in the two cases



# Diphones – a different kind of unit

---

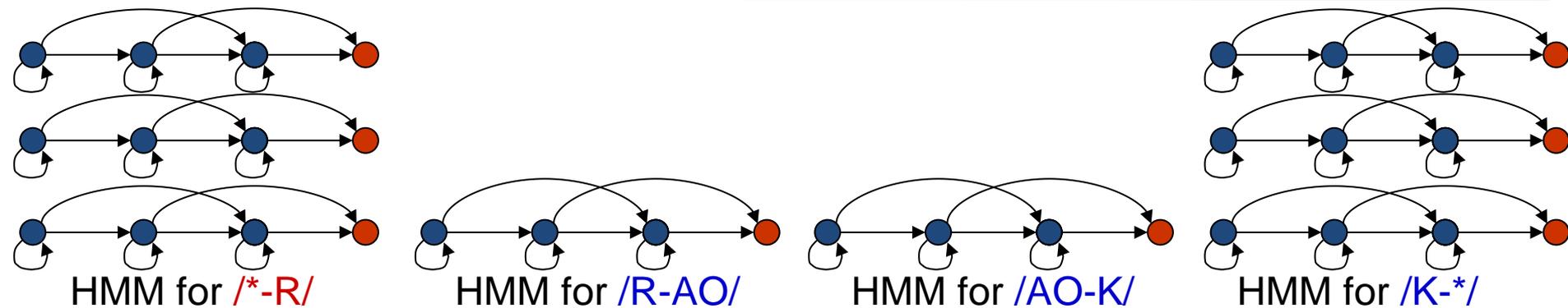
- A diphone begins at the center of one phoneme and ends at the center of the next phoneme
- Diphones are much less affected by contextual, or co-articulation effects than phonemes themselves
  - All regions of the diphone are consistent, i.e. they all provide evidence for the identity of the diphone
  - Boundary regions represent loci of phonemes and are consistent
  - Central regions represent transitions between consistent loci, and are consistent
- Consequently, diphones are much better building blocks for word HMMs than phonemes
- For a language with  $N$  phonemes, there are  $N^2$  diphones
  - These will require correspondingly larger amounts of training data
  - However, the actual number of sub-word units remains limited and enumerable
    - As opposed to words that are unlimited in number

# The Diphone

---

- Phonetic representation of ROCK:
  - ROCK: R AO K
- Diphone representation:
  - ROCK: (??-R), (R-AO), (AO-K), (K-??)
  - Each unit starts from the middle of one phoneme and ends at the middle of the next one
- **Word boundaries are a problem**
  - The diphone to be used in the first position (??-R) depends on the last phoneme of the previous word!
    - ?? is the last phoneme of the previous word
  - Similarly, the diphone at the end of the word (K-??) depends on the next word
- We build a separate diphone-based word model for every combination of preceding and following phoneme observed in the grammar
- ***The boundary units are SHARED by adjacent words***

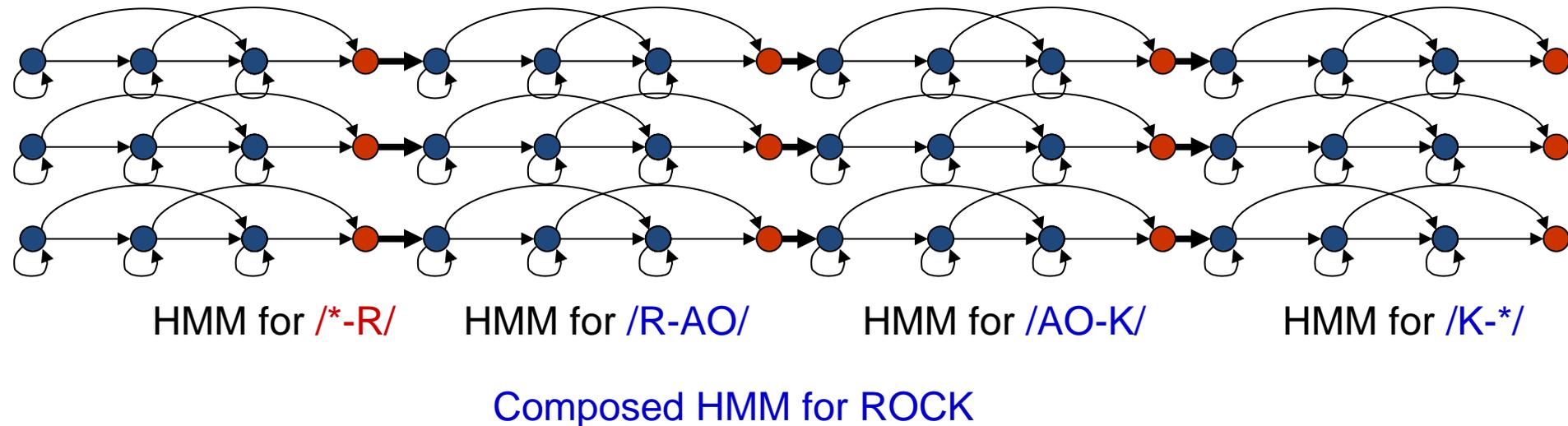
# Building word HMMs with diphones



Components of HMM for ROCK

- Dictionary entry for ROCK: /R/ /AO/ /K/
- Word boundary units are not unique
- The specific diphone HMMs to be used at the ends of the word depend on the previous and succeeding word
  - E.g. The first diphone HMM for ROCK in the word series JAILHOUSE ROCK is /S-R/, whereas for PLYMOUTH ROCK, it is /TH-R/
- As a result, there are as many HMMs for “ROCK” as there are possible left-neighbor, right-neighbor phoneme combinations

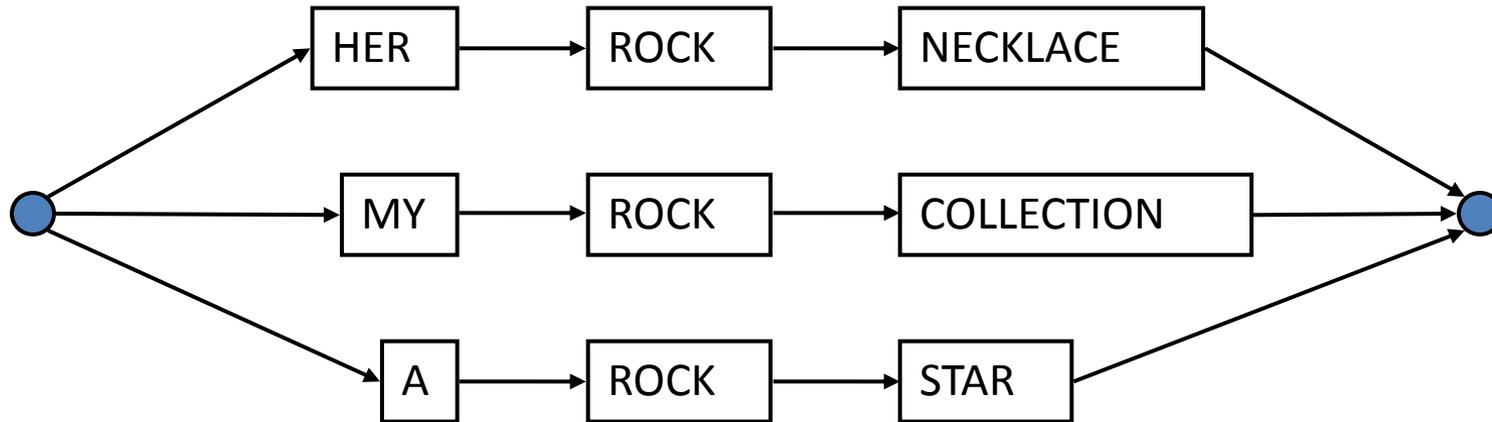
# Building word HMMs with diphones



- We end up with as many word models for ROCK as the number of possible combinations of words to the right and left

# Diphone Networks

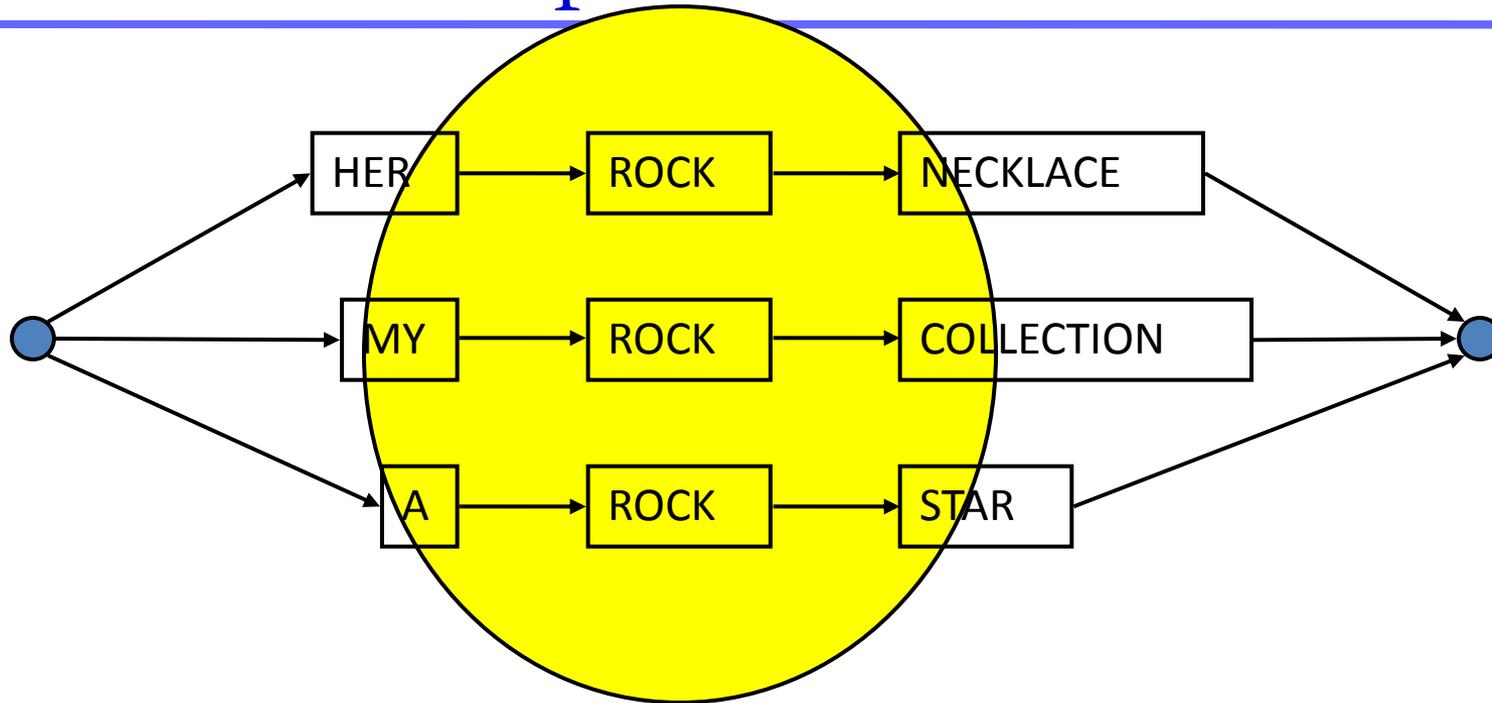
---



- Consider this example grammar

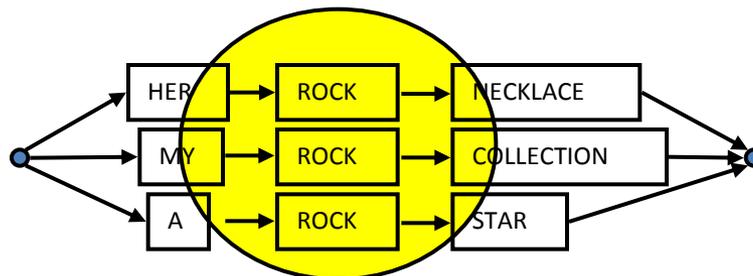
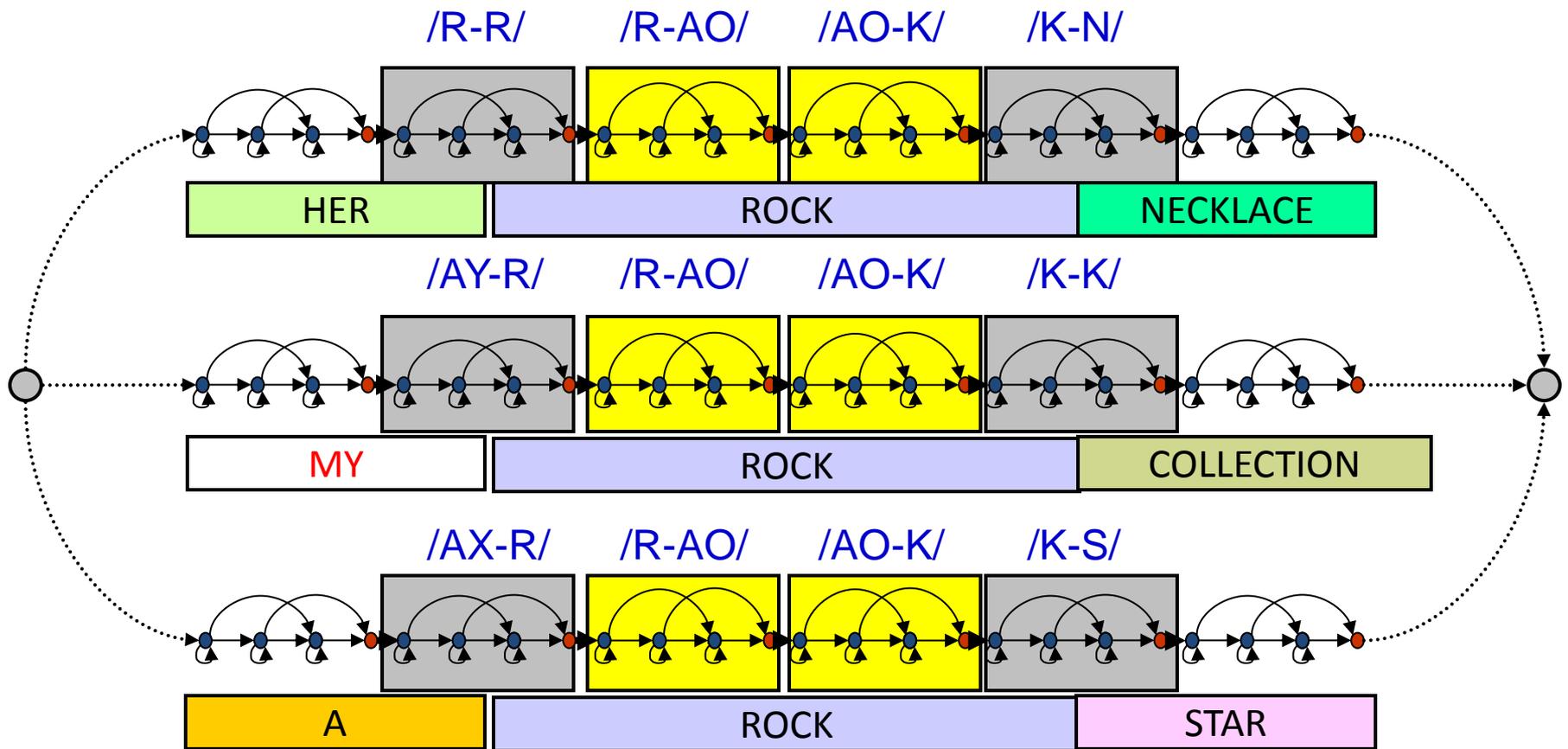
# Diphone Networks

---

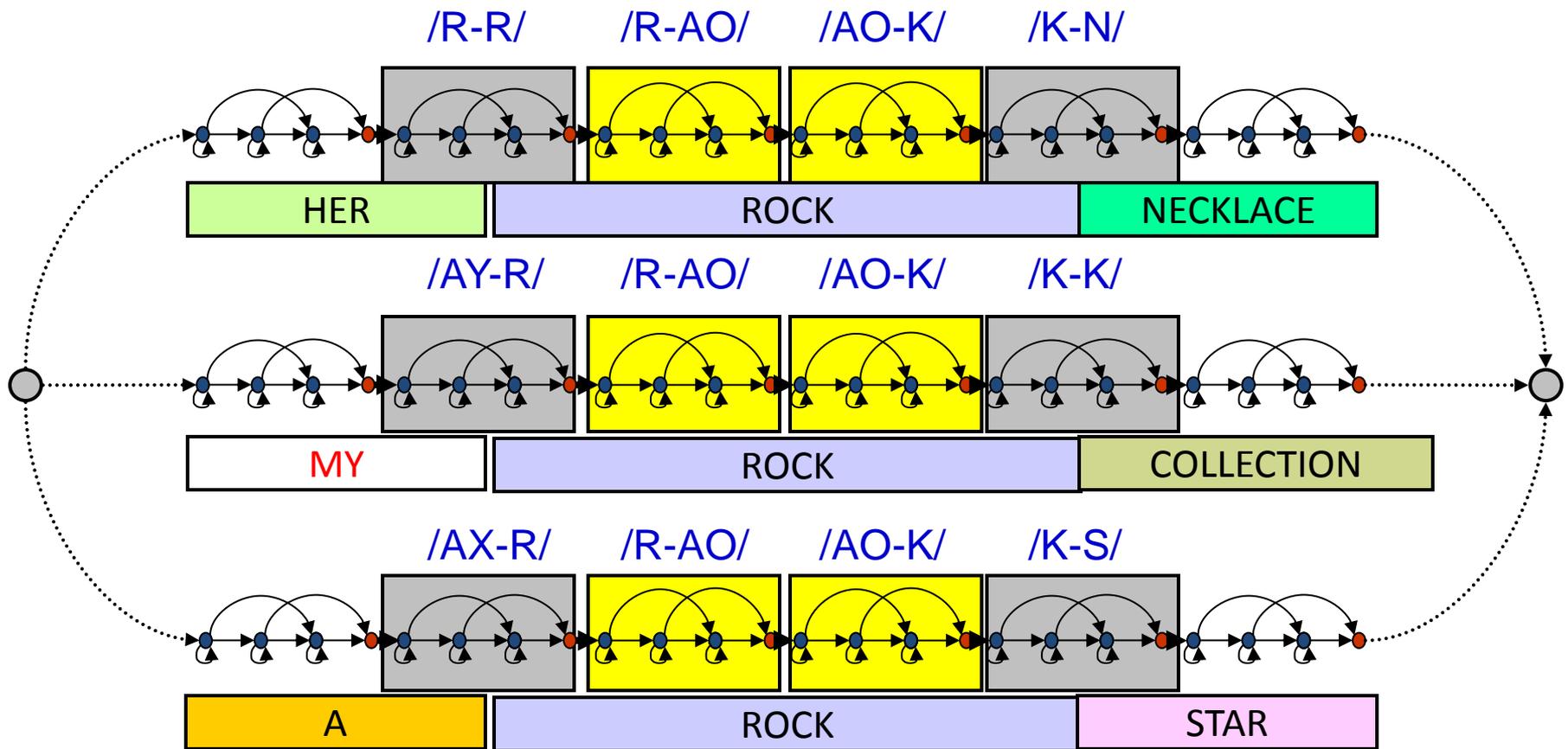


- Consider this example grammar
- For illustration, we will concentrate on this region

# Building word HMMs with diphones

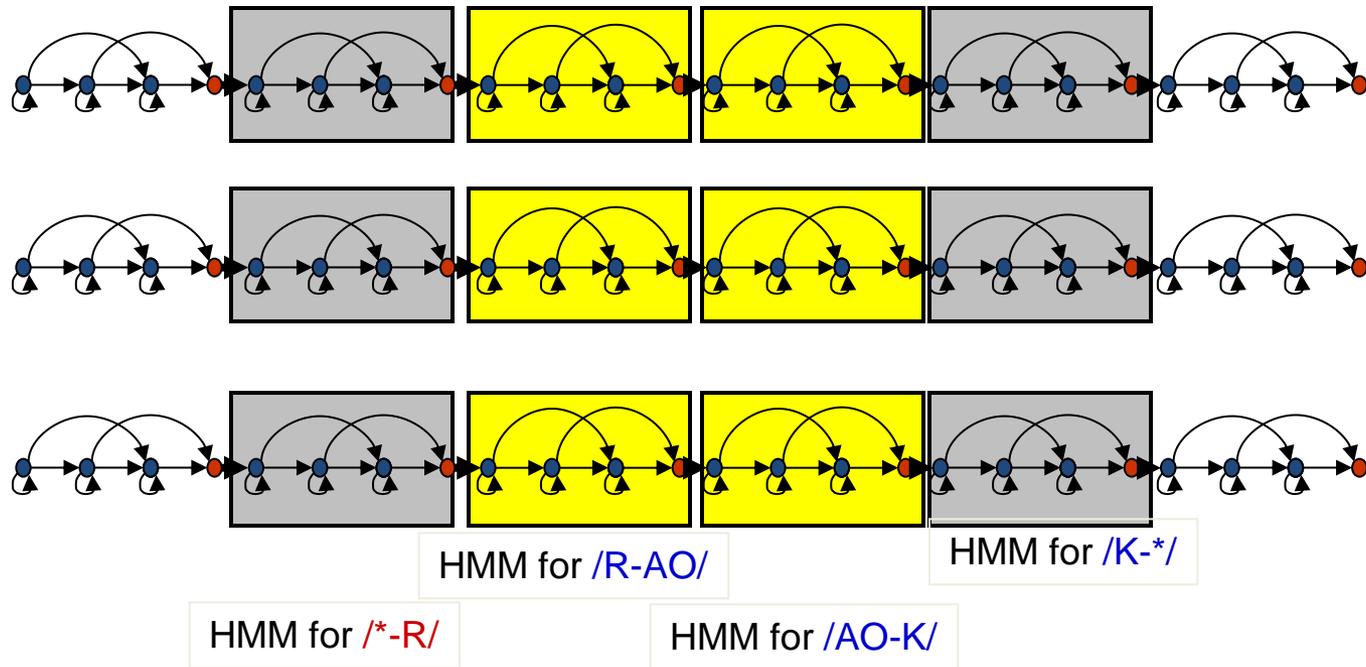


# Building word HMMs with diphones



- Each instance of Rock is a different model
  - The 3 instances are not copies of one another
- The boundary units (gray) are shared with adjacent words

# Building word HMMs with diphones

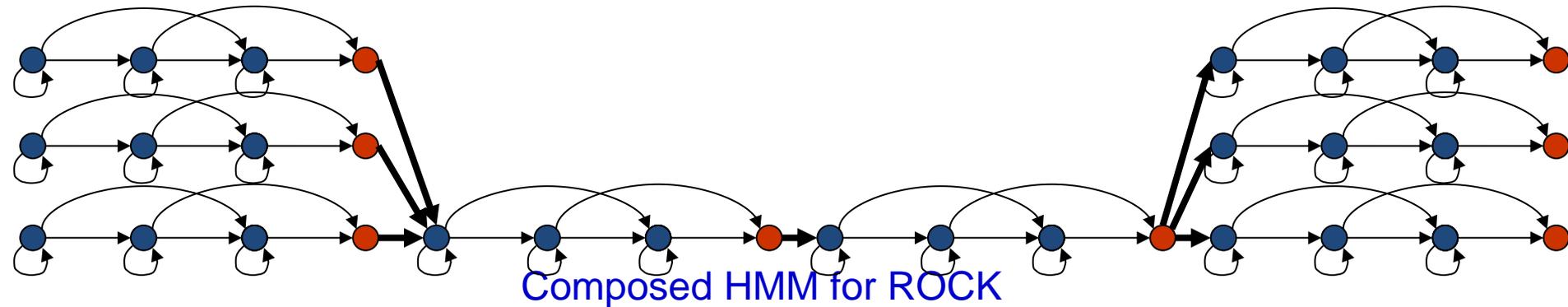


Composed HMM for ROCK

- We end up with as many word models for ROCK as the number of possible words to the right and left

# Building word HMMs with diphones

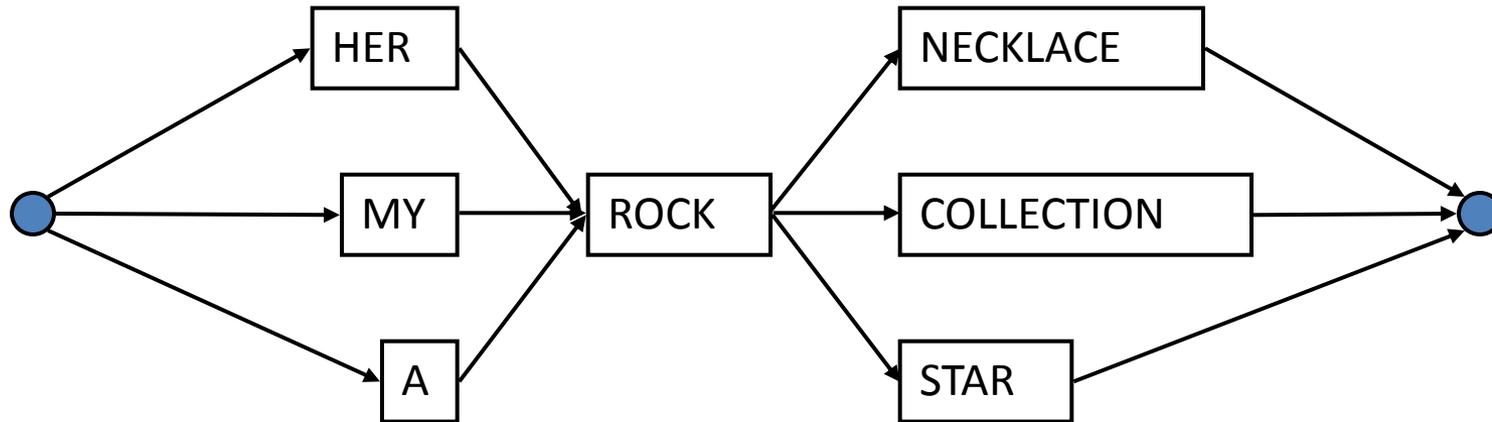
---



- Under some conditions, portions of multiple versions of the model for a word can be collapsed
  - Depending on the grammar

# Diphone Networks

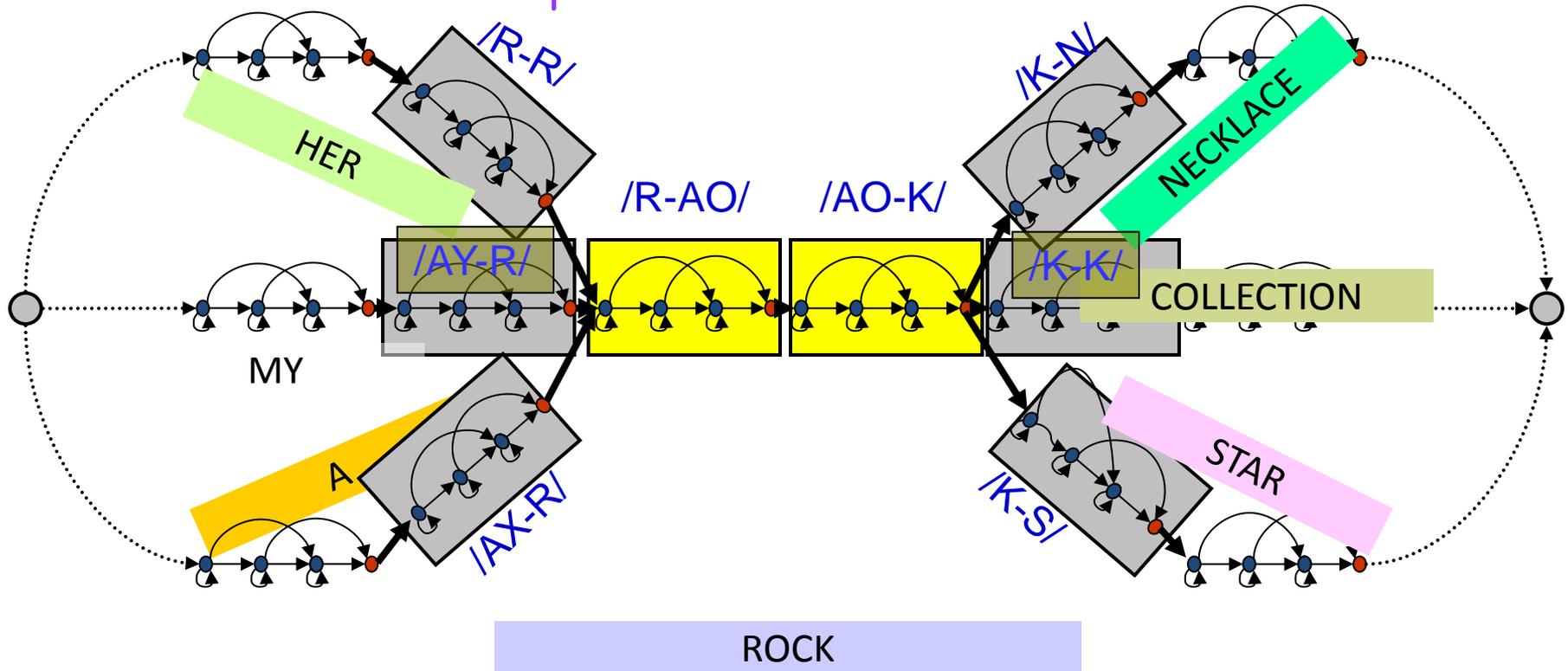
---



- Consider this example grammar

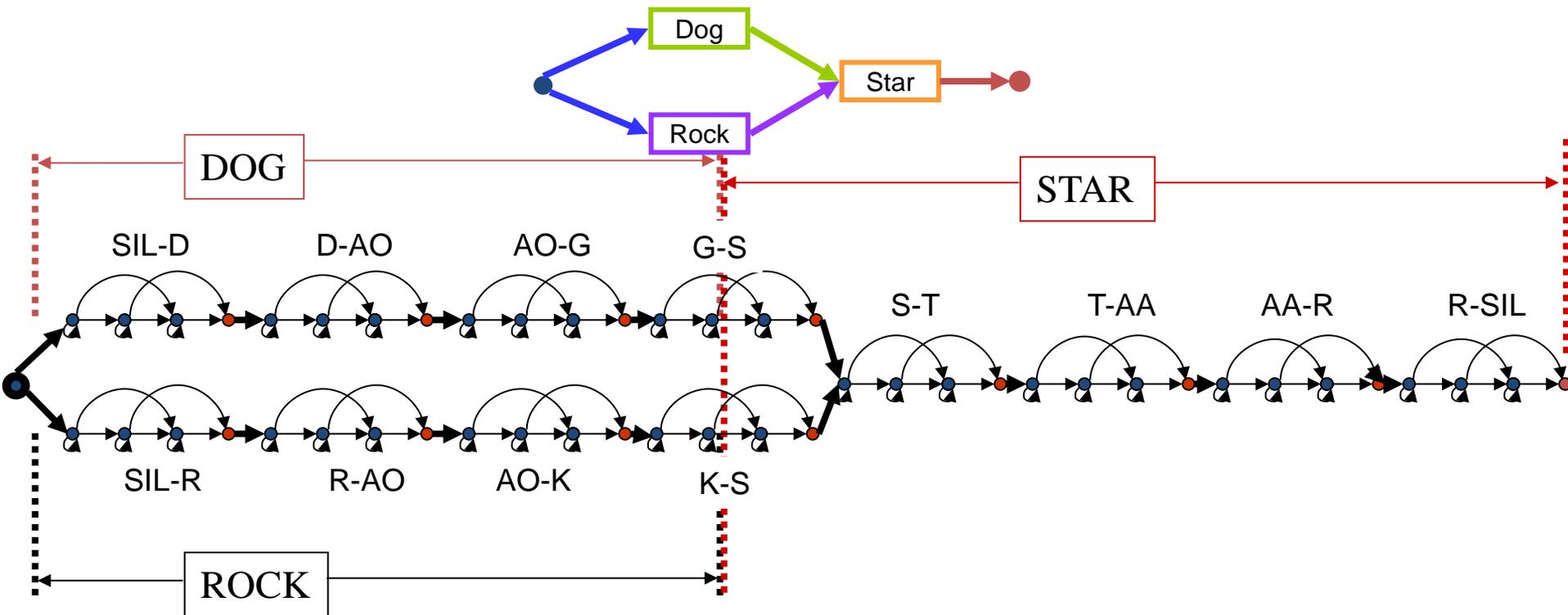
# Building word HMMs with diphones

Connections between HMMs to be built  
as explained earlier



- As before, the transition diphones belong to both words in each case
- Such collapsing of the central word is possible only when they all represent the same entry in the grammar

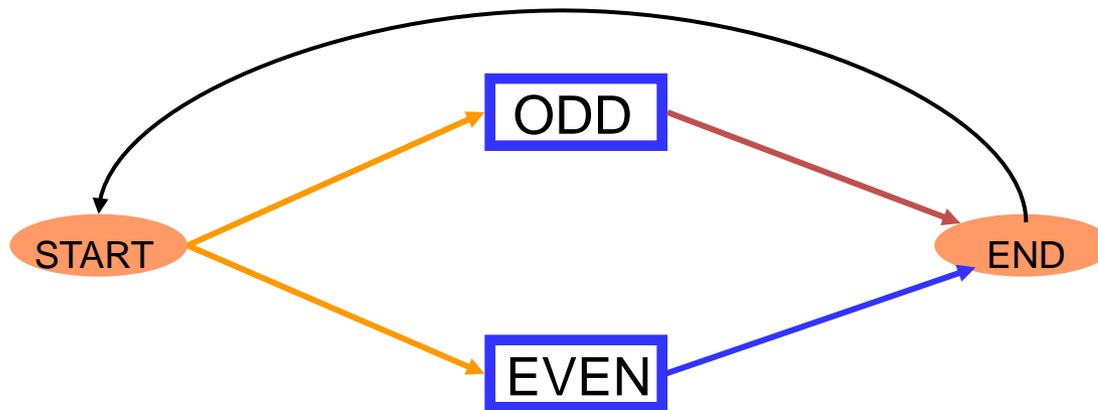
# A simple example of a complete grammar



- One way of forming a Diphone-based HMM for the simple grammar that recognizes “DOG STAR” or “ROCK STAR”
- The boundary units (G-S and K-S) are actually shared by two words

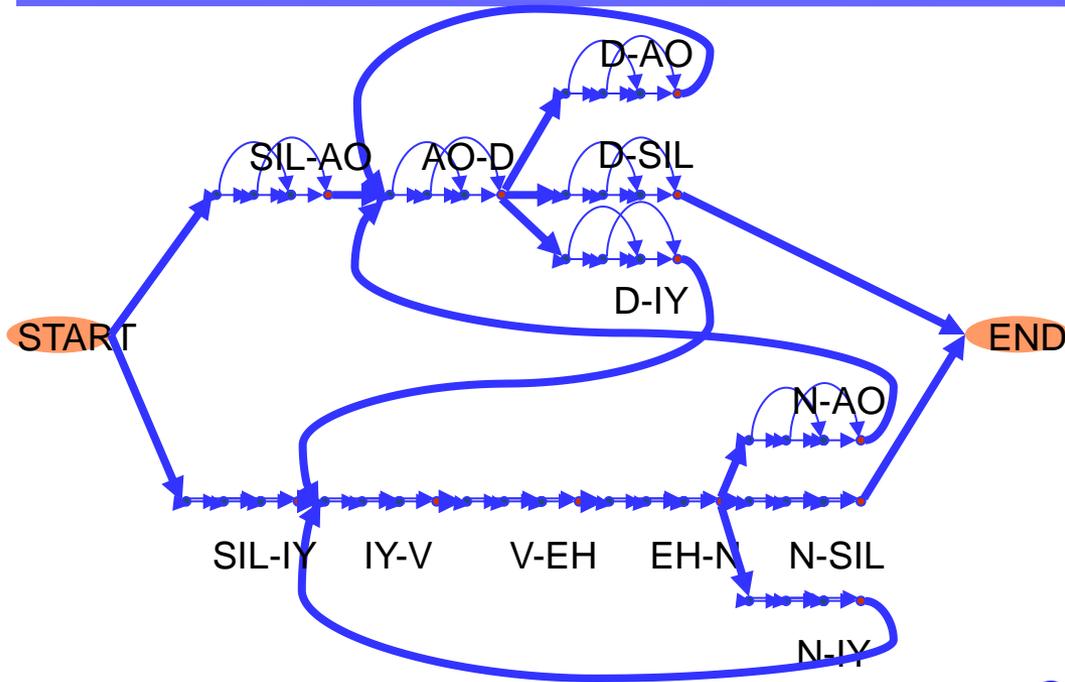
# Another Example

---

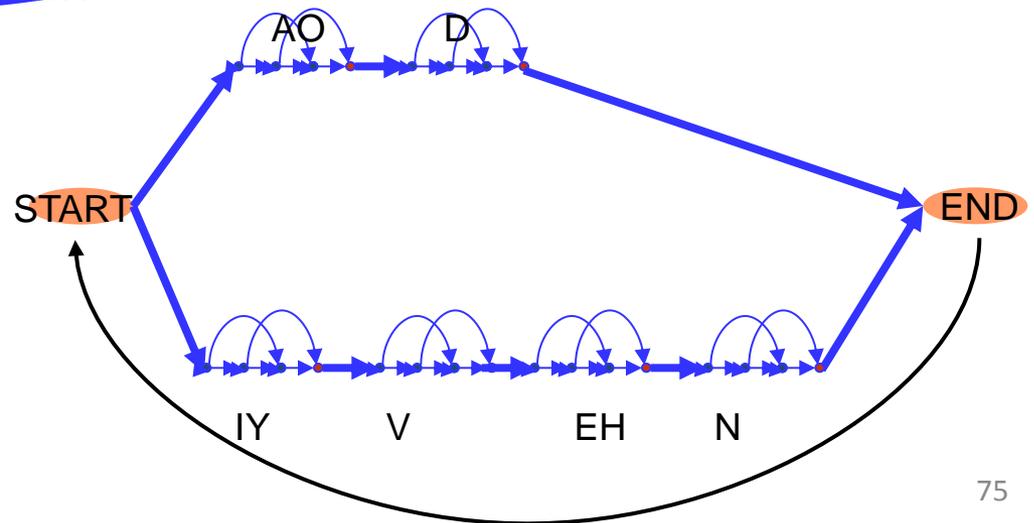


A simple grammar that allows only the words EVEN and ODD. The probability of ODD following ODD is different from that of ODD following EVEN etc.

# The Full Diphone-based HMM



The Diphone HMM (left) is much more complex than the corresponding phoneme-based HMM (below)



# Building a Diphone-based recognizer

---

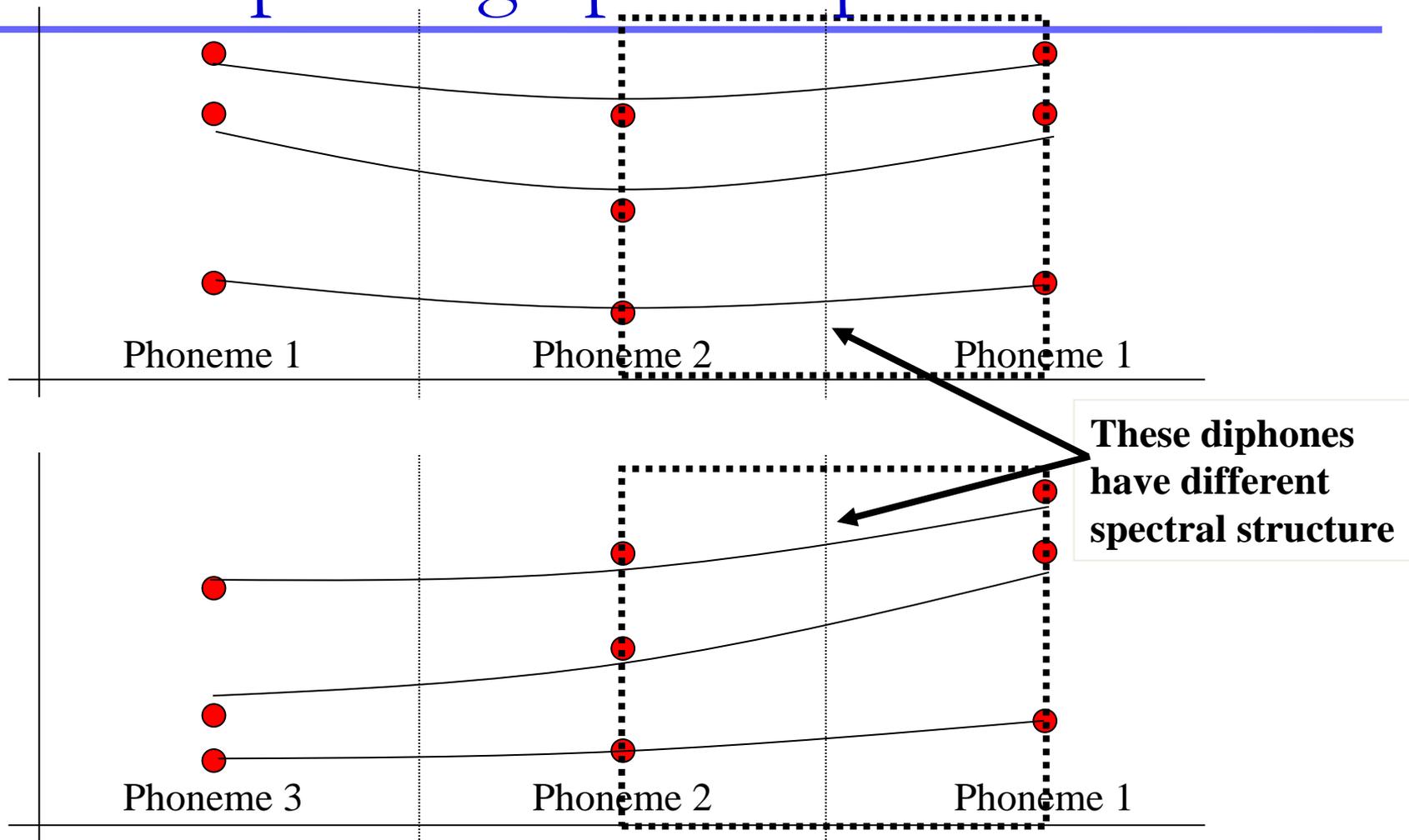
- Train models for all Diphones in the language
  - If there are  $N$  phonemes in the language, there will be  $N^2$  diphones
  - For 40 phonemes, there are thus 1600 diphones; still a manageable number
- Build the HMM for the Grammar using Diphone models
  - There will no longer be distinctly identifiable word HMM because boundary units are shared among adjacent words

# Word-boundary Diphones

---

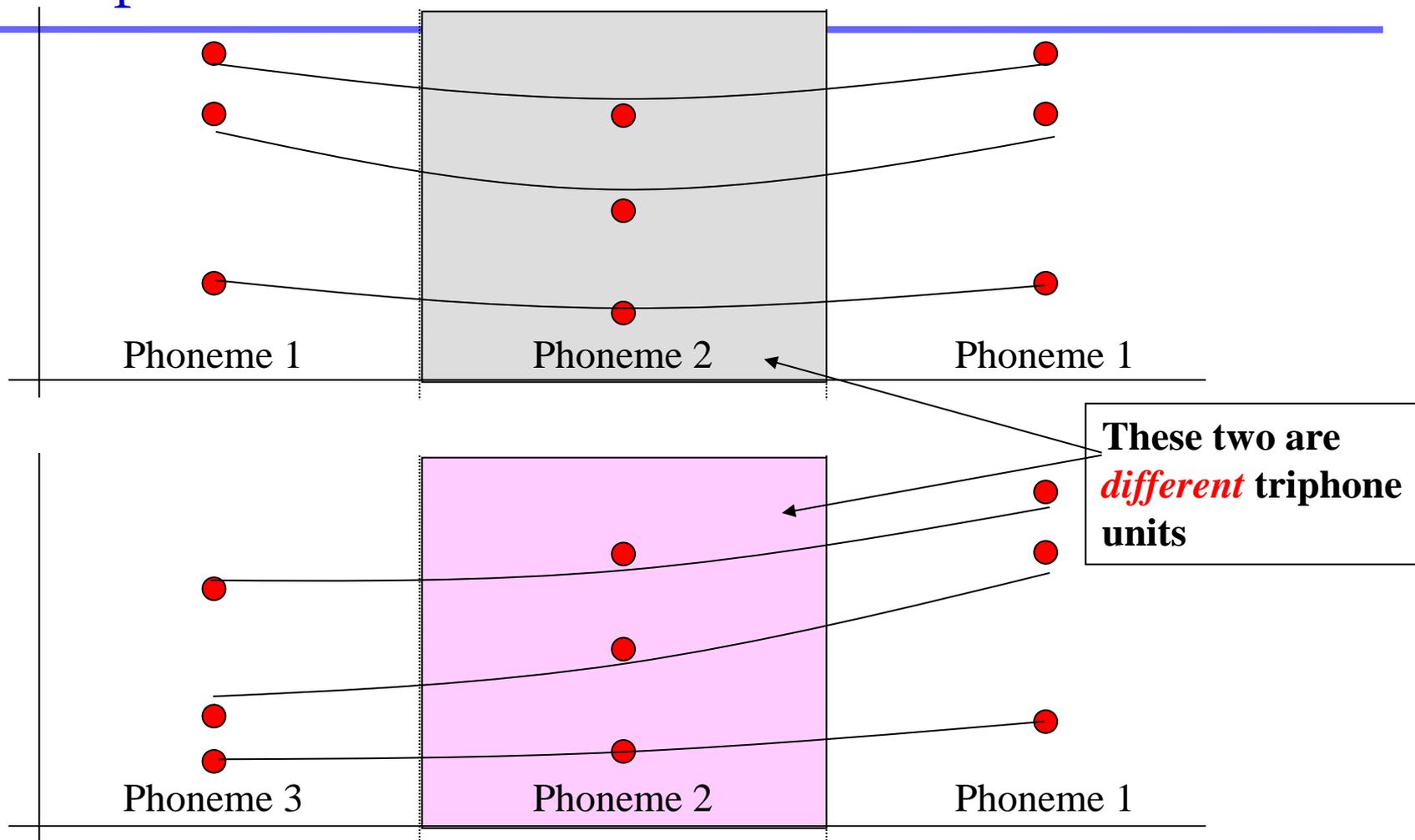
- Cross-word diphones have somewhat different structure than within-word diphones, even when they represent the same phoneme combinations
  - E.g. the within-word diphone AA-F in the word “AFRICA” is different from the word-boundary diphone AA-F in BURKINA FASO”
  - Stresses applied to different portions of the sounds are different
  - This is true even for cross-syllabic diphones
- It is advantageous to treat cross-word diphones as being distinct from within-word diphones
  - This results in a doubling of the total number of diphones in the language to  $2N^2$ 
    - Where N is the total number of phonemes in the language
  - We train cross-word diphones only from cross-word data and within-word diphones from only within-word data
- The separation of within-word and word-boundary diphones does not result in any additional complication of the structure of sentence HMMs during recognition

# Improving upon Diphones



- Loci may never occur in fluent speech
  - Resulting in variation even in Diphones
- The actual spectral trajectories are affected by adjacent phonemes
  - Diphones do not capture all effects

# Triphones: PHONEMES IN CONTEXT



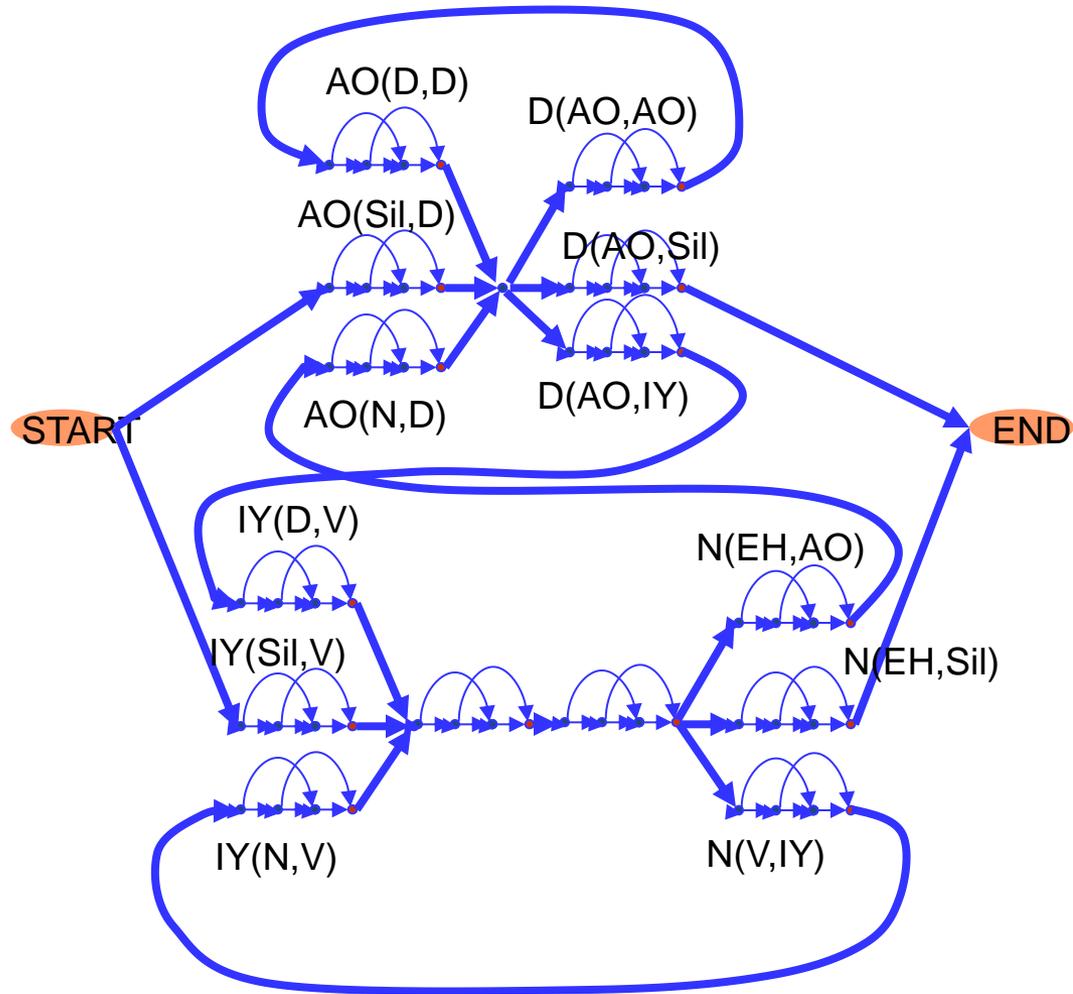
- Triphones represent phoneme units that are specific to a context
  - E.g. “The kind of phoneme 2 that follows phoneme 3 and precedes phoneme 3”
- **The triphone is *not* a triplet of phonemes – it still represents a single phoneme**

# Triphones

---

- Triphones are actually *specialized phonemes*
  - The triphones AX (B, T), AX( P, D), AX(M,G) all represent variations of AX
- To build the HMM for a word, we simply concatenate the HMMs for individual triphones in it
  - E.g R AO K : R(??, AO) AO (R,K) K(AO,??)
  - We link the HMMs for the triphones for R(??,AO), AO(R,K) and K(AO,??)
- Unlike diphones, no units are shared across words
- Like diphones, however, the boundary units R(??,AO), K(AO,??) depend on the boundary phonemes of adjacent words
  - In fact it becomes more complex than for diphones

# The Triphone-based HMM



# Cross-word triphones complicate HMMs

- Triphones at word boundaries are dependent on neighbouring words.
- This results in significant complication of the HMM for the language (through which we find the best path, for recognition)
  - Resulting in larger HMMs and slower search

## Dictionary

Five:            **F AY V**  
Four:            **F OW R**  
Nine:            **N AY N**  
<sil>:            **SIL**  
++breath++: **+breath+**

Listed here are five “words” and their pronunciations in terms of “phones”. Let us assume that these are the only words in the current speech to be recognized. The recognition vocabulary thus consists of five words. The system uses a **dictionary** as a reference for these mappings.

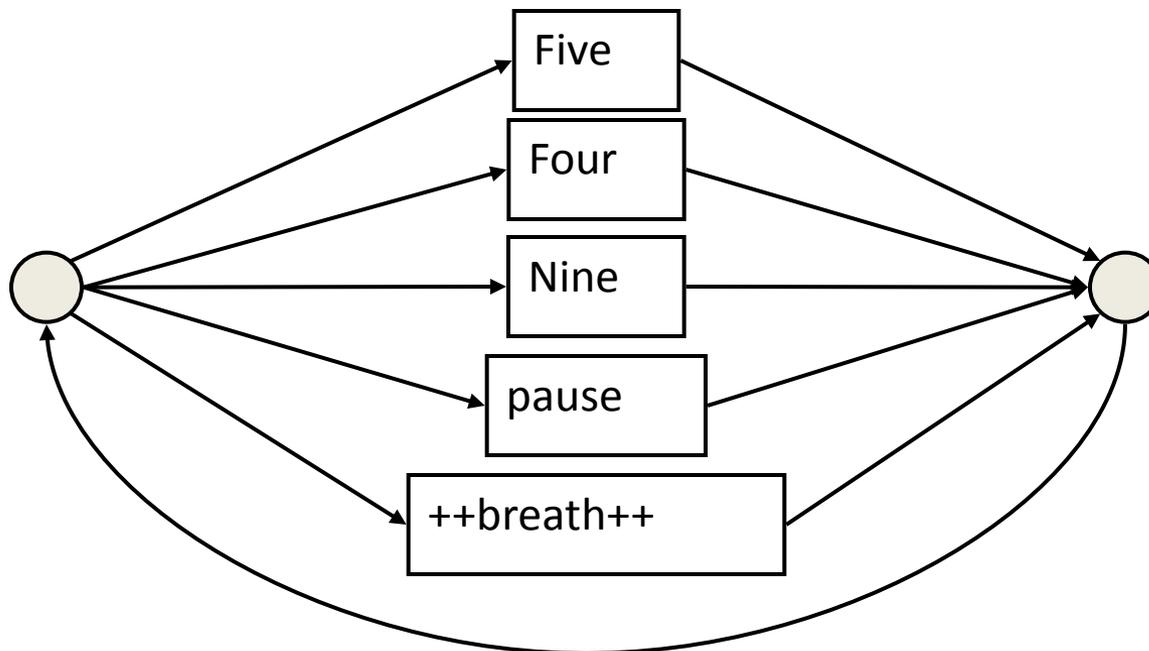
## Lexicon

Five			
Four			
Nine			
<sil>			
++Breath++			

	= F
	= AY
	= V
	= OW
	= R
	= N
	= SIL
	= +breath+

# A Slightly More Complex Grammar for Illustration

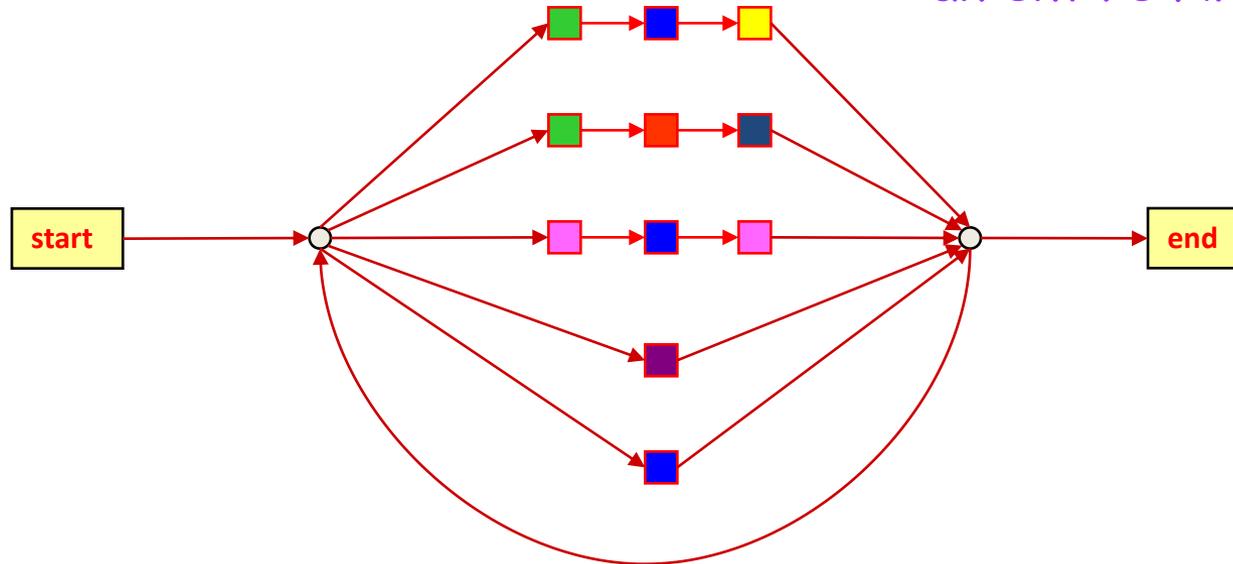
---



- We will compose an HMM for the simple grammar above
  - A person may say any combination of the words five, nine and four with silences and breath inbetween

# Using (CI) Phonemes

Each coloured square represents an entire HMM



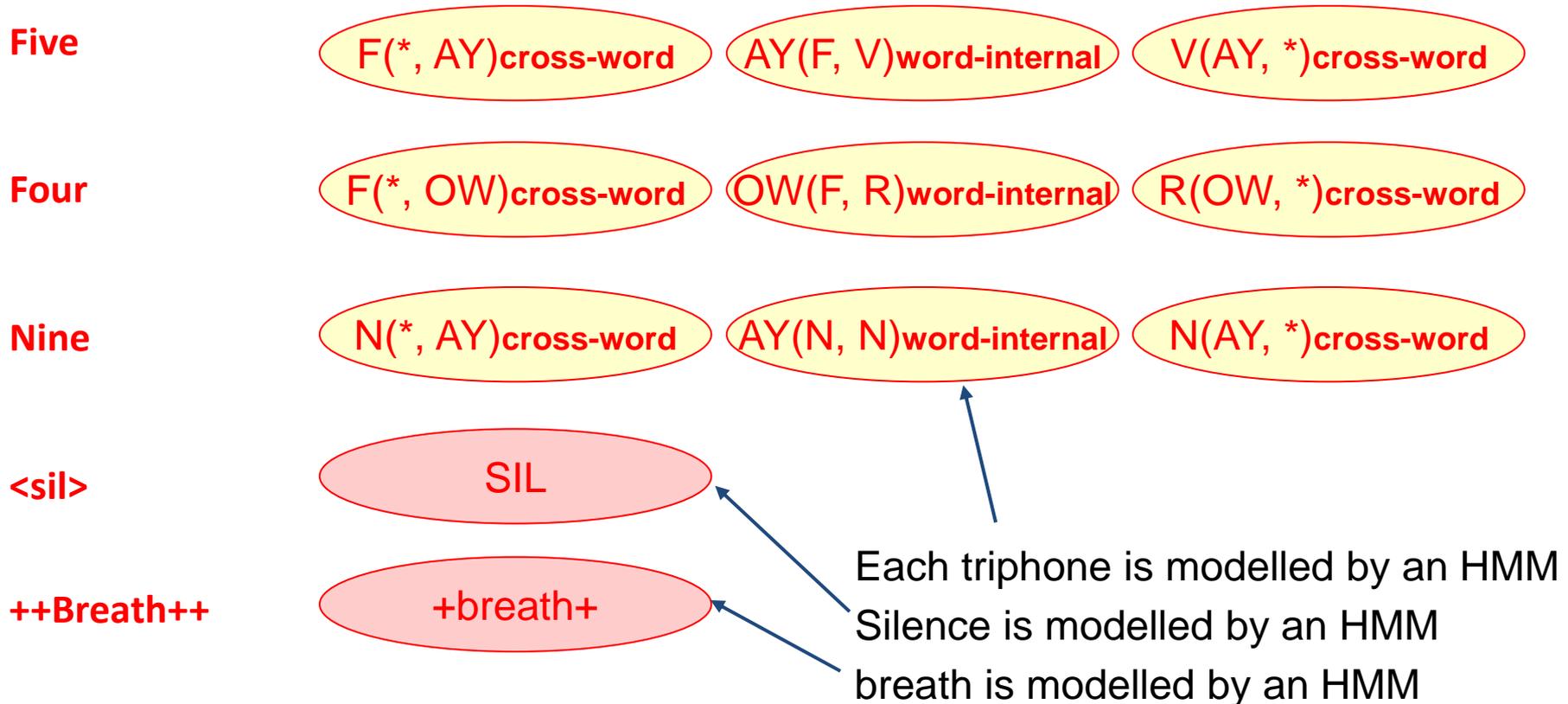
Lexicon	
Five	Green Blue Yellow
Four	Green Red Dark Blue
Nine	Pink Blue Pink
<sil>	Dark Purple
++Breath++	Blue

Word boundary units are not context specific.  
All words can be connected to (and from) null nodes

Green	= F
Blue	= AY
Yellow	= V
Red	= OW
Dark Blue	= R
Pink	= N
Dark Purple	= SIL
Blue	= +breath+

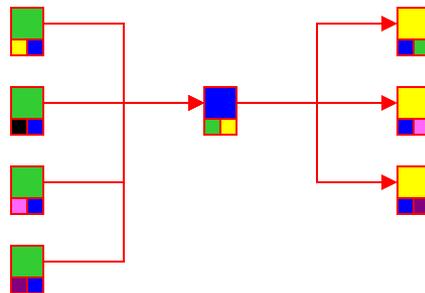
# Using Triphones

Using the dictionary as reference, the system first maps each word into triphone-based pronunciations. Each triphone further has a characteristic **label or type**, according to where it occurs in the word. *Context is not initially known for cross-word triphones.*



# Using Triphones

HMM for "Five".  
This is composed of  
8 HMMs.



Each triple-box represents a triphone. Each triphone model is actually a left-to-right HMM.

## Lexicon

Five	  
Four	  
Nine	  
<sil>	
++Breath++	

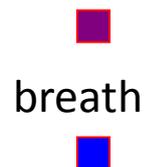
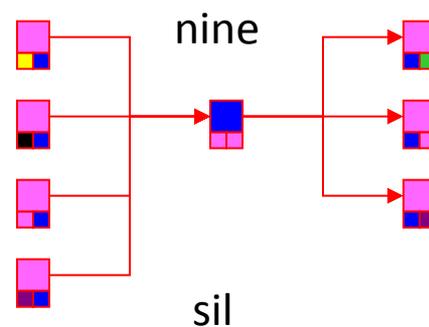
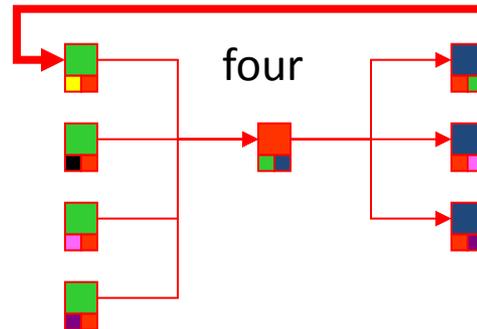
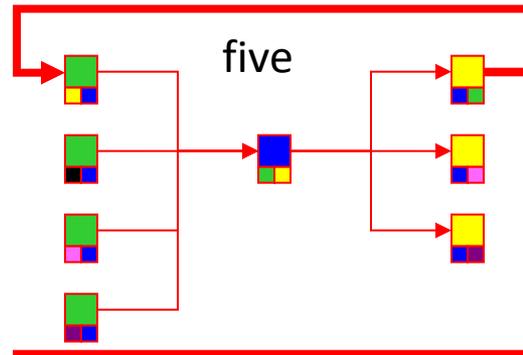
	= F
	= AY
	= V
	= OW
	= R
	= N
	= SIL
	= +breath+

**A triphone is a *single* context-specific phoneme. It is *not* a sequence of 3 phones.**

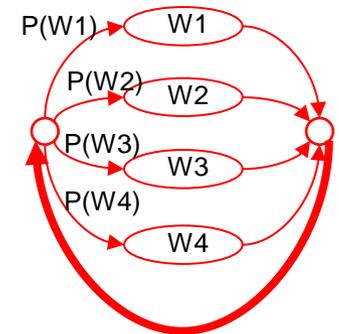
Expand the word Five

- All last phones (except +breath+) become left contexts for first phone of Five.
- All first phones (except +breath+) become right contexts for last phone of Five
- Silence can form contexts, but itself does not have any context dependency.
- **Filler** phones (e.g. +breath+) are treated as silence when building contexts. Like silence, they themselves do not have any context dependency.

# The triphone based Grammar HMM

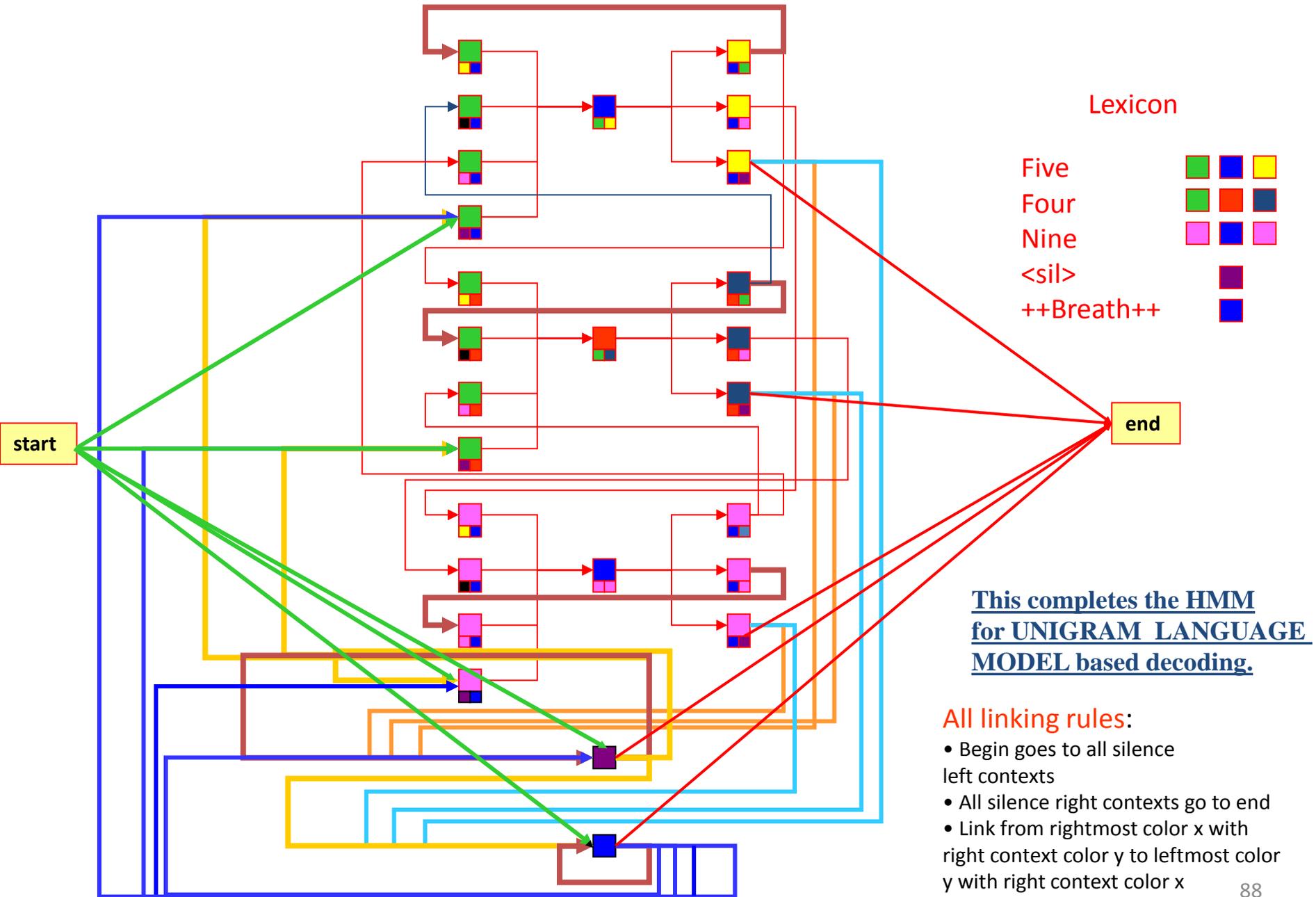


Lexicon	
Five	■ ■ ■
Four	■ ■ ■
Nine	■ ■ ■
<sil>	■
++Breath++	■



**Linking rule:**  
Link from rightmost color  $x$  with right context color  $y$  to leftmost color  $y$  with right context color  $x$

# The triphone based Grammar HMM

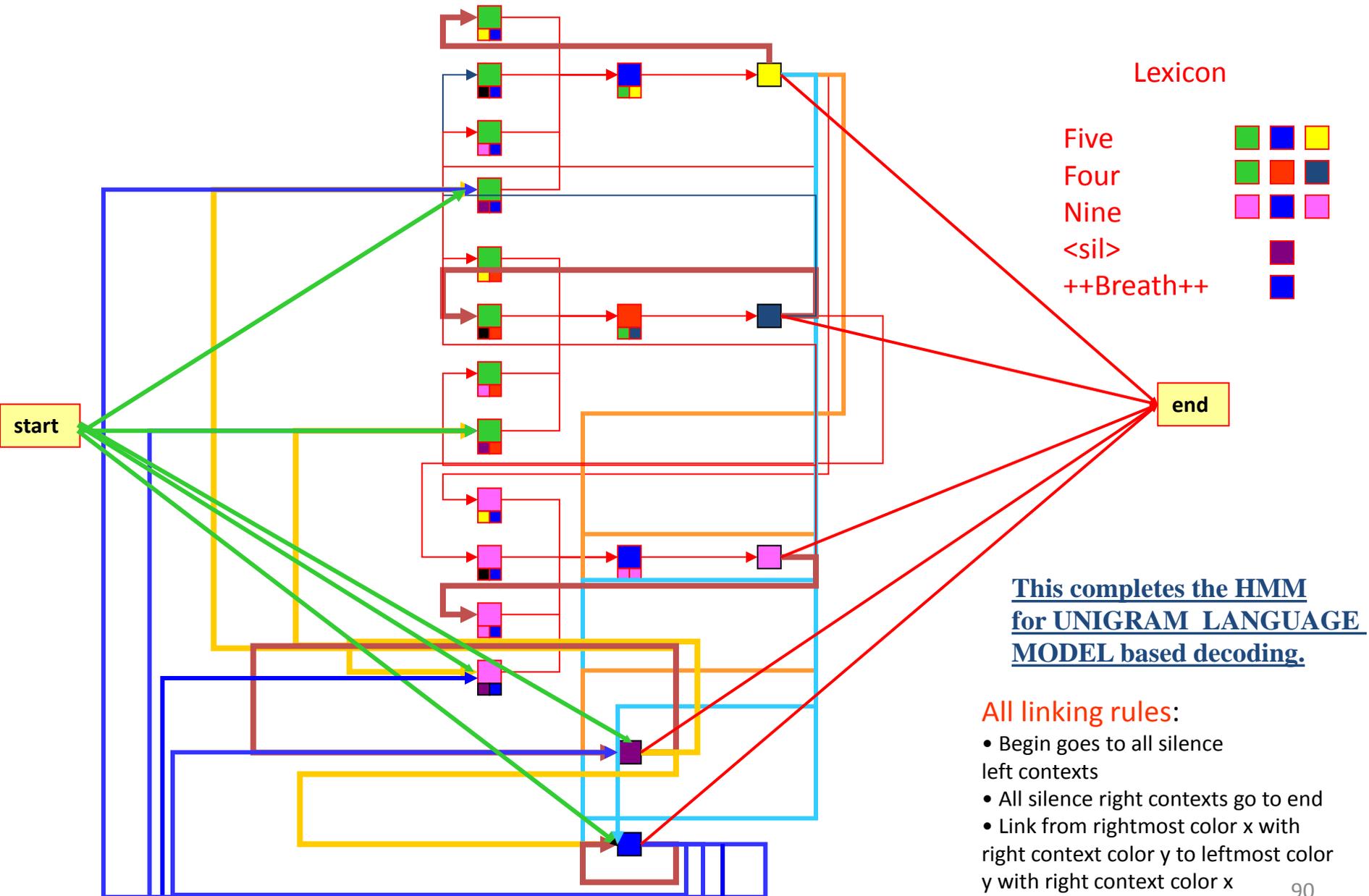


# The triphone based Grammar HMM

---

- Even a simple looping grammar becomes very complicated because of cross-word triphones
- The HMM becomes *even more* complex when some of the words have only one phoneme
  - There will be as many instances of the HMM for the word as there are word contexts
  - No portion of these HMMs will be shared!
- We can simplify this structure by using CI phonemes at either word entry or word exit or both
  - This is acceptable since triphones are specialized cases of phonemes
- Typically this reduces the number of states, but not transitions
- Approximations will result in reduction of accuracy
  - Approximating word-exit triphones with CI phonemes is less harmful than approximating word-entry triphones

# CI Phones at the exit



# Types of triphones

---

- A triphone in the middle of a word sounds different from the same triphone at word boundaries
  - e.g the word-internal triphone AX(G,T) from GUT: G AX T
  - Vs. cross-word triphone AX(G,T) in BIG ATTEMPT
- The same triphone in a single-word (e.g when the central phoneme is a complete word) sounds different
  - E.g. AX(G,T) in WAG A TAIL
  - The word A: AX is a single-phone word and the triphone is a single-word triphone
- We distinguish four types of triphones:
  - Word-internal
  - Cross-word at the beginning of a word
  - Cross-word at the end of a word
  - Single-word triphones
- Separate models are learned for the four cases and appropriately used

# Context Issues

---

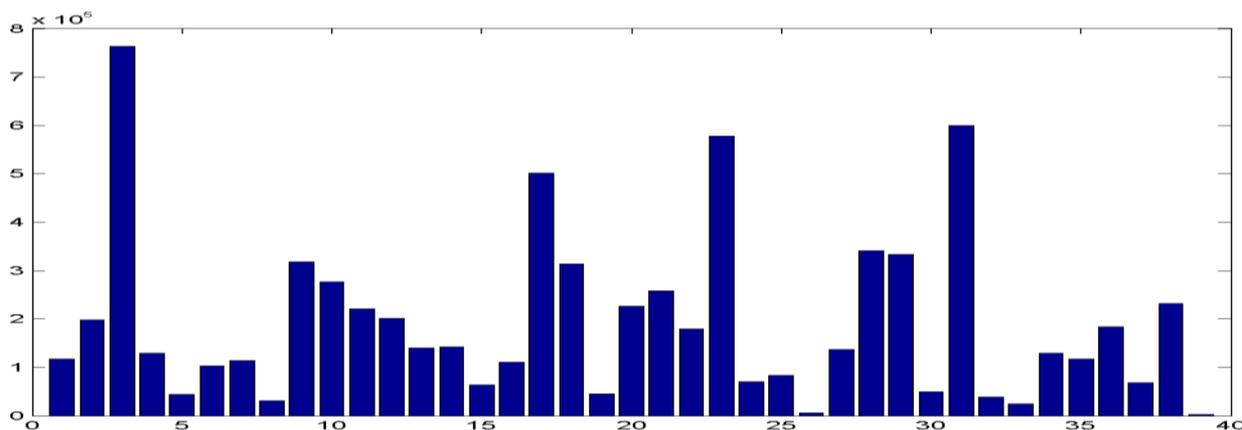
- Phonemes are affected by adjacent phonemes.
- If there is no adjacent phoneme, i.e. if a phoneme follows or precedes a silence or a pause, that will also have a unique structure
- We will treat “silence” as a context also
- “Filler” sounds like background noises etc. are typically treated as equivalent to silence for context
- “Voiced” filler sounds, like “UM”, “UH” etc. do affect the structure of adjacent phonemes and must be treated as valid contexts

# Training Data Considerations for Nphone Units

---

- 1.53 million words of training data (~70 hours)
- All 39 phonemes are seen (100%)
- 1387 of 1521 possible diphones are seen (91%)
  - Not considering cross-word diphones as distinct units
- 24979 of 59319 possible triphones are seen (42%)
  - not maintaining the distinction between different kinds of triphones

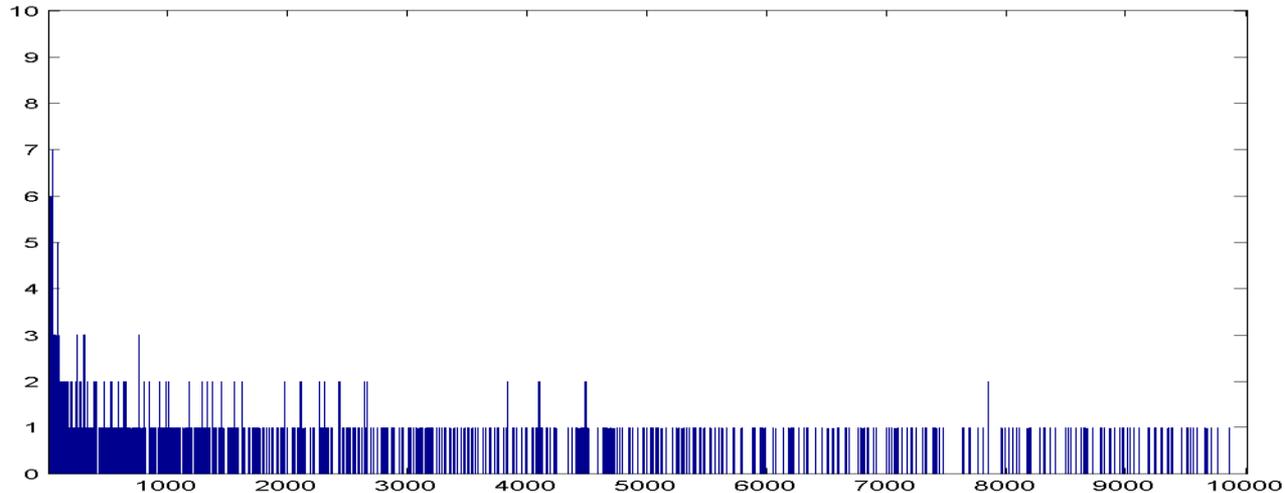
# Counts of CI Phones



Histogram of the number of occurrences of the 39 phonemes in 1.5 million words of Broadcast News

- All context-independent phonemes occur in sufficient numbers to estimate HMM parameters well
  - Some phonemes such as “ZH” may be rare in some corpora. In such cases, they can be merged with other similar phonemes, e.g. ZH can be merged with SH, to avoid a data insufficiency problem

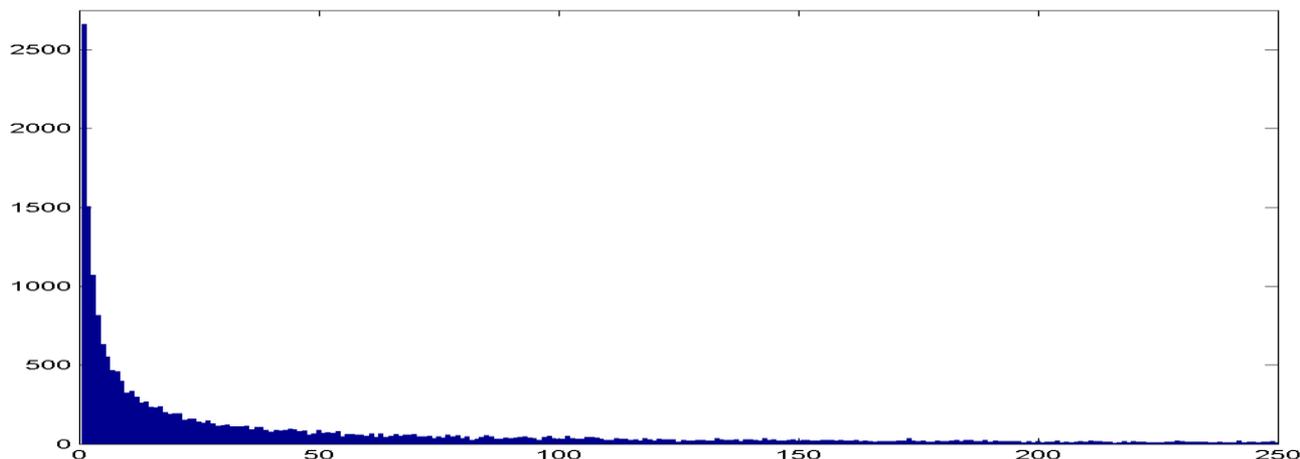
# Counts of Diphones



“Count of counts” histogram for the 1387 diphones in 1.5 million words of Broadcast News

- Counts (x axis)  $> 10000$  not shown, Count of counts (Y axis) clipped from above at 10
- The figure is much flatter than the typical trend given by Zipf’s law
- The mean number of occurrences of diphones is 4900
- Most diphones can be well trained
  - Diphones with low (or 0) count, e.g. count  $< 100$ , account for less than 0.1% of the total probability mass
    - Again, contrary to Zipf’s law
  - Diphones with low count, or unseen diphones, can be clustered with other similar diphones with minimal loss of recognition performance

# Counts of Triphones



"Count of counts" histogram for the 24979 triphones in 1.5 million words of Broadcast News

- Counts (x axis)  $> 250$  not shown
- Follows the trends of Zipf's law very closely
  - Because of the large number of triphones
- The mean number of occurrences of *observed* triphones is 300
- The majority of the triphones are not seen, or are rarely seen
  - 58% of all triphones are never seen
  - 86% of all triphones are seen less than 10 times
  - The majority of the triphones in the language will be poorly trained
    - Back off to CI phones or use parameter sharing techniques for these

# Higher order Nphones

---

- Spectral trajectories are also affected by farther contexts
  - E.g. by phonemes that are two phonemes distant from the current one
- The effect of longer contexts is much smaller than that of immediate context, in most speech
  - The use of longer context models only results in relatively minor improvements in recognition accuracy
- There are far too many possibilities for longer context units
  - E.g, there are  $40^5$  possible quinphone units (that consider a 2-phoneme context on either side), even if we ignore cross-word effects
- Cross-word effects get far more complicated with longer-context units
  - Cross-word contexts may now span multiple words. The HMM for any word thus becomes dependent on the the previous two (or more) words, for instance.
  - Even a simple unigram graph may end up having the same complexity as a trigram LM graph, as a result

# Training Tradeoffs

---

- Word models
  - Very effective, if they can be well trained
  - Difficult to train well, when vocabularies get large
  - Cannot recognize words that are not seen in training data
- Context-independent phoneme models
  - Simple to train; data insufficiency rarely a problem
  - All phonemes are usually seen in the training data
    - If some phonemes are not seen, they can be mapped onto other, relatively common phonemes
- Diphones
  - Better characterization of sub-word acoustic phenomena than simple context-independent phonemes
  - Data insufficiency a relatively minor problem. Some diphones are usually not seen in training data.
    - Their HMMs must be inferred from the HMMs for seen diphones

# Training Tradeoffs

---

- Triphones
  - Characterizations of phonemes that account for contexts on both sides
  - Result in better recognition than diphones
  - Data insufficiency is a problem: the majority of triphones are not seen in training data.
    - Parameter sharing techniques must be used to train uncommon triphones derive HMMs for unseen triphones
  - Advantage: can back-off to CI phonemes when HMMs are not available for the triphones themselves
- Higher order Nphone models
  - Better characterizations than triphones, however the benefit to be obtained from going to Nphones of higher contexts is small
  - Data insufficiency a huge problem – most Nphones are not seen in the training data
    - Complex techniques required for *inferring* the models for unseen Nphones
    - Alternatively, one can backoff to triphones or CI-phonemes

# Recognition Tradeoffs

---

- Word models
  - Very effective for recognition, when vocabularies are small
    - Poor training affects large vocabulary systems
  - Cannot recognize words that are not seen in the training data
- Phoneme and Nphone models: words that are never seen in the training data can still be recognized
  - Their HMMs can be composed from the HMMs for the phonetic units
- Context-independent phoneme models
  - Results in very compact HMMs and fast decoding speeds
  - Relatively poor performance
- Diphones
  - There are many more Diphones than CI phonemes
    - The total number of HMM parameters to be stored is larger as a result
  - Cross-word effects complicate sentence HMM structure
  - The resulting sentence HMMs are larger than those obtained with CI phonemes
  - Recognition with diphones is slower but more accurate than recognition with CI phonemes

# Recognition Tradeoffs

---

- Triphones
  - Much larger in number than diphones
    - Requires the storage of a larger number of HMM components
  - Cross-word effects complicate sentence HMM structure
  - Triphone-based systems are larger, slower, and more accurate than those based on diphones or CI phonemes
- Higher order Nphones
  - Even larger in number than triphones
  - Cross-word effects are more complicated than for diphones or triphones
  - Sentence HMM graphs become prohibitive in size
- What to use
  - Word models are best when the vocabulary is small (e.g. digits).
  - CI phoneme based models are rarely used
  - Where accuracy is of prime importance, triphone models are usually used
  - If reduced memory footprint and speed are important, e.g. in embedded recognizers, diphone models are often used
  - Higher-order Nphone models are rarely used in the sentence HMM graphs used for recognition. However, they are frequently used for *rescoring*
    - Rescoring will be discussed in a future lecture.

# Summary

---

- Have discussed subword units and the need for them
- Building word HMMs using subword units
- Context dependent subword units
  - Diphones
  - Triphones
  - Issues
- Tradeoffs
- Next: More details