LECTURE 37: LEXICAL TREES

- Objectives:
  - A typical speech recognition architecture
  - Complexity of N-gram decoding
  - Definition of a lexical tree
  - Basic computational issues

This lecture follows the course textbook closely:

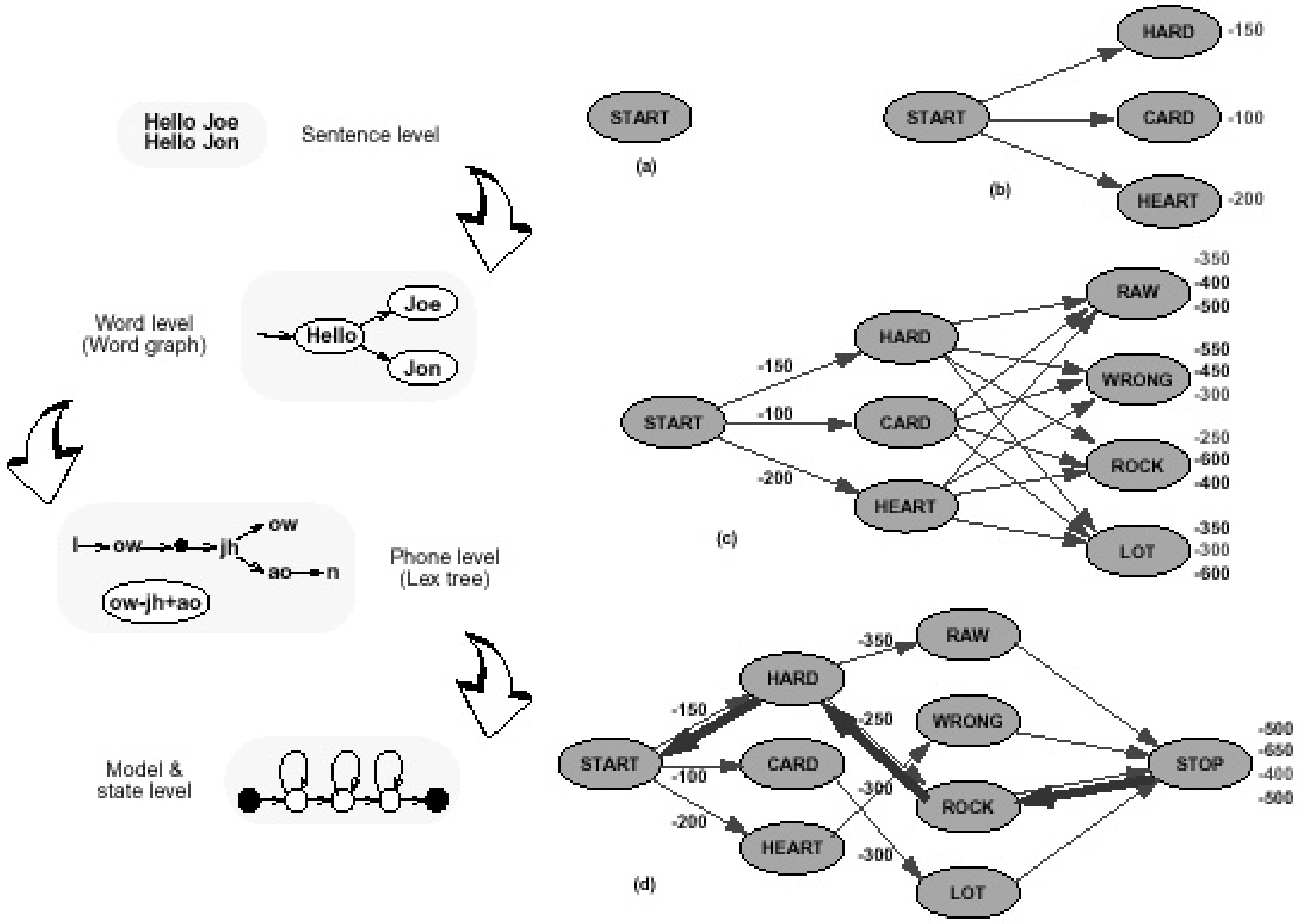
> X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

Another good source for some of this information is:

> N. Deshmukh, A. Ganapathiraju and J. Picone, "Hierarchical Search for Large Vocabulary Conversational Speech Recognition," *IEEE Signal Processing Magazine*, vol. 16, no. 5, pp. 84-107, September 1999.

A TYPICAL SPEECH RECOGNITION SYSTEM

- A typical system decomposes sentences into words, words into phone sequences, and phones into HMM models:

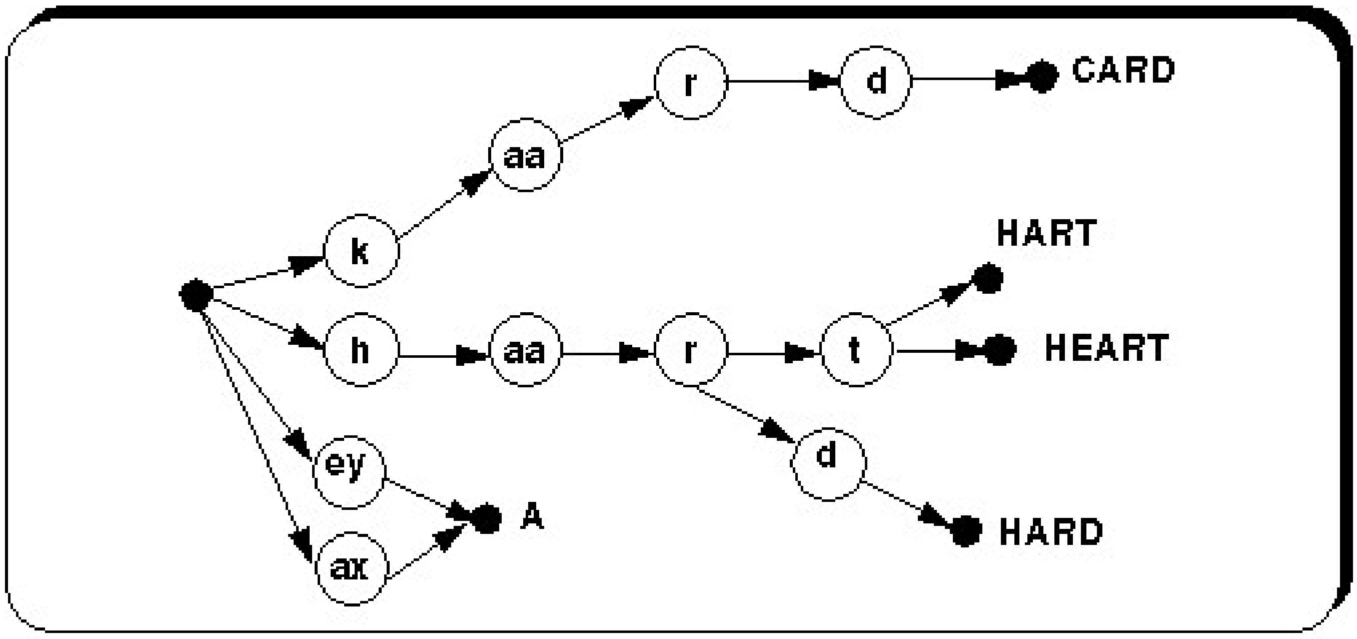


COMPLEXITY OF N-GRAM SEARCH

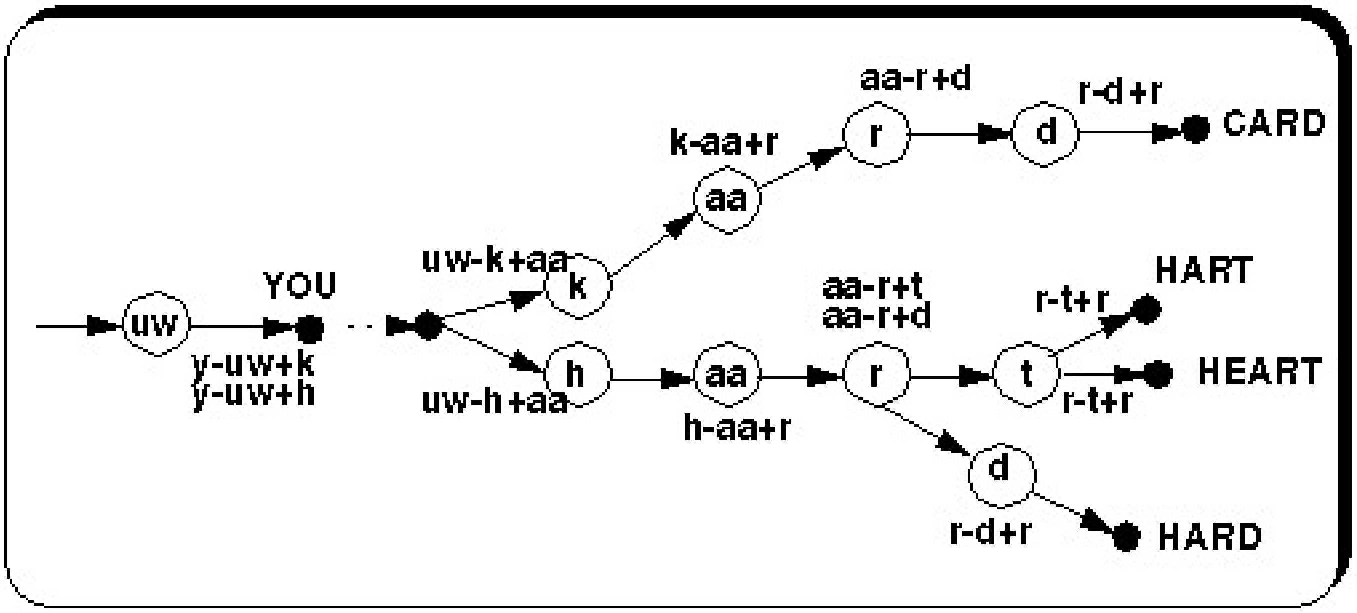
- Although it is always desirable to use as many knowledge sources as possible, there are practical problems integrating such information into a time-synchronous search.
- One alternate strategy is to use a mult-pass search. However, the more accurate the first-pass, the better the performance on subsequent passes.
- One of the most critical parts of search is the **tree lexicon**.
- In a **linear lexicon**, each word is represented as a linear sequence of phonemes independent of other words. For example, though *task* and *tasks* share the same root, we do not share any of their history during the search process.
- Large lexicons introduce enormous complexity for a backoff N-gram language model because we must "start" all words in the lexicon for every unique history in our current search space.
- Can we share some of the underlying phonetic structure of words in the lexicon?

LEXICAL TREES

- A **lexical tree** is a data structure that allows us to share histories between words in the lexicon:

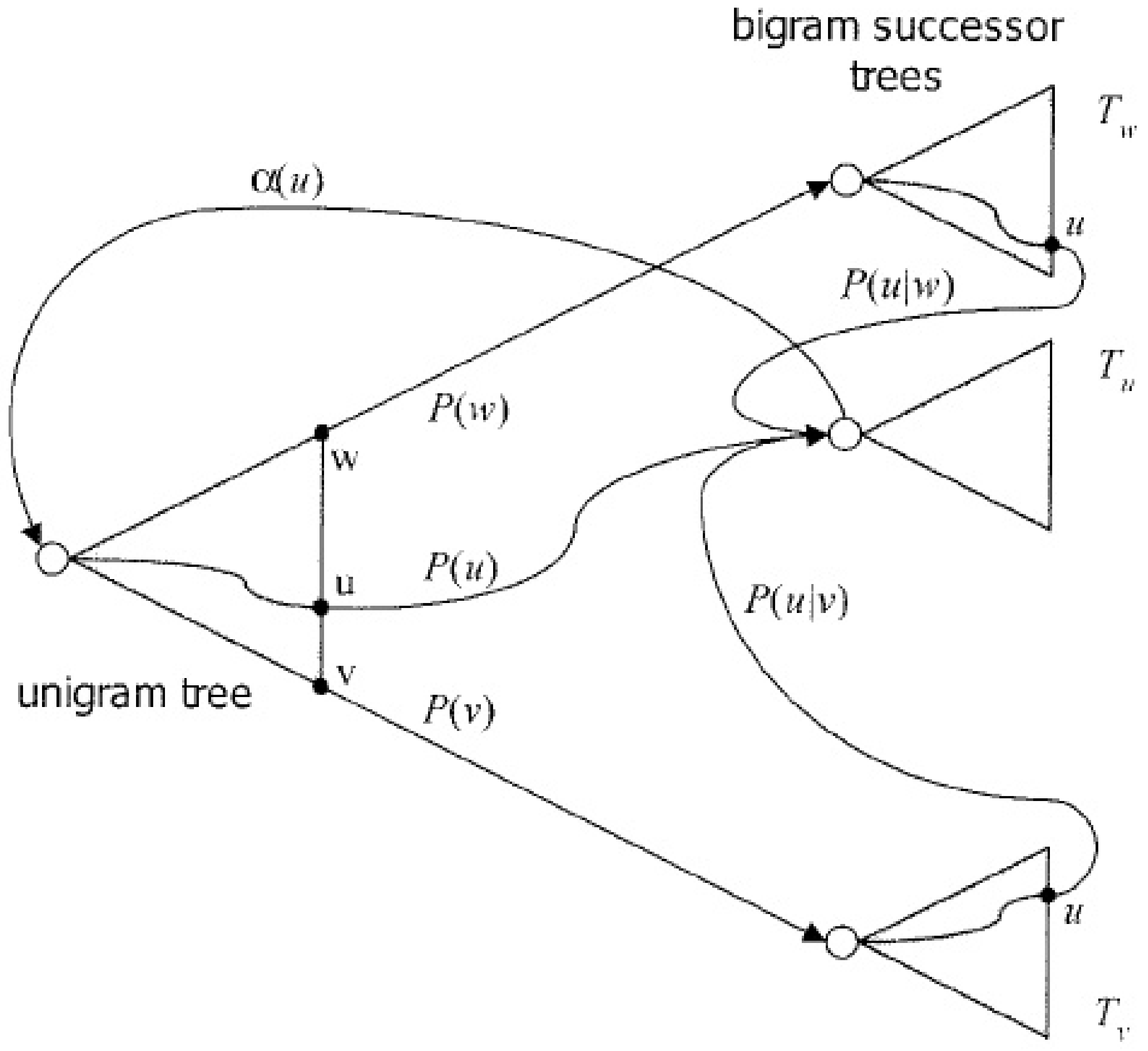


- Most systems use some form of dynamic context expansion:



MULTIPLE COPIES OF PRONUNCIATION TREES

- A simple lexical tree is sufficient if no language model is used.
- For higher-order N-gram models, the linguistic state cannot be determined locally. History plays an important role. For example, for bigrams, a tree copy is required for each predecessor word.
- Efficient pruning can eliminate most of the unnecessary tree copies.
- To deal with tree copies, you can create redundant trees for each word context. However, this is expensive.
- Many of the active state hypotheses correspond to the same redundant unigram state because the language model probability cannot be applied until the next word has been observed.
- A successor tree approach can be used to eliminate this redundancy:



- However, for backoff language models, this isn't as efficient as it might seem (unless aggressive pruning is used to allow only a small subset of words to follow a given word).

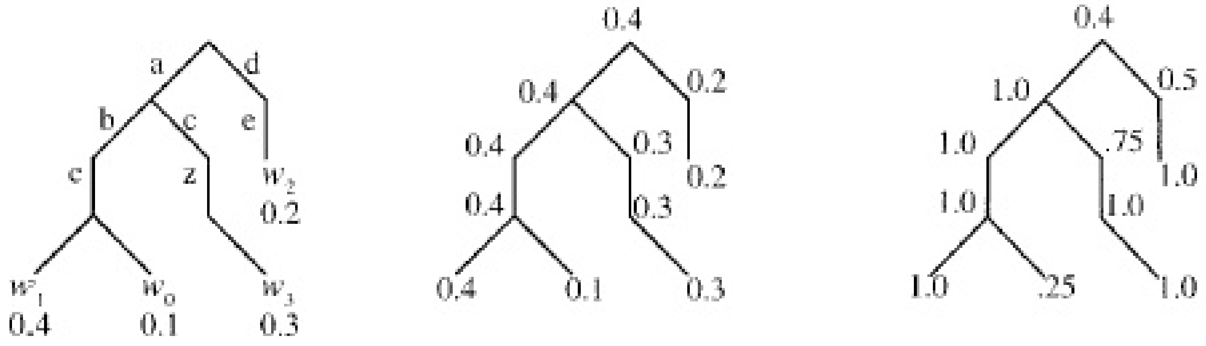
FACTORED LANGUAGE MODEL PROBABILITIES

- Premise: We can improve accuracy and minimize resources if we can apply the language model score as soon as possible.
- Factoring LM probabilities across a tree is one such idea:

$$P^*(n) = \max_{x \in child(n)} P(x)$$

$$F^*(n) = \frac{P^*(n)}{P^*(parent(n))}$$

- We can embed these probabilities in the pronunciation tree:



- In practice, such ideas require strong emphasis on the language model to be effective (such as WSJ or NAB). Applications such as conversational speech, in which acoustic ambiguity is extreme, and acoustic scores tend to dominate decoding, do not benefit as much from such approaches.