[Return to Main](#)

[Objectives](#)

Statistical Models:
[Noisy Channel Model](#)
[Wheel of Fortune](#)
[Word Prediction](#)

Syntactic Constraints:
[State Machine](#)
[Ad Hoc Approaches](#)
[Networks](#)

Formal Language:
[Rewrite Rules](#)
[Chomsky Hierarchy](#)

On-Line Resources:
[HLTSurvey](#)
[Statistical Methods in NLP](#)
[Software: SRILM](#)
[Software: CMUSLM](#)

LECTURE 29: FORMAL LANGUAGE THEORY

- Objectives:
  - Communication Theoretic Approach
  - Chomsky Hierarchy
  - Network Grammars
  - Production Rules

This lecture combines material from the course textbook:

> 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.

and from this source:

> F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, Boston, Massachusetts, USA, ISBN: 0-262-10066-5, 1998.

LECTURE 29: FORMAL LANGUAGE THEORY

- Objectives:
  - Communication Theoretic Approach
  - Chomsky Hierarchy
  - Network Grammars
  - Production Rules

This lecture combines material from the course textbook:

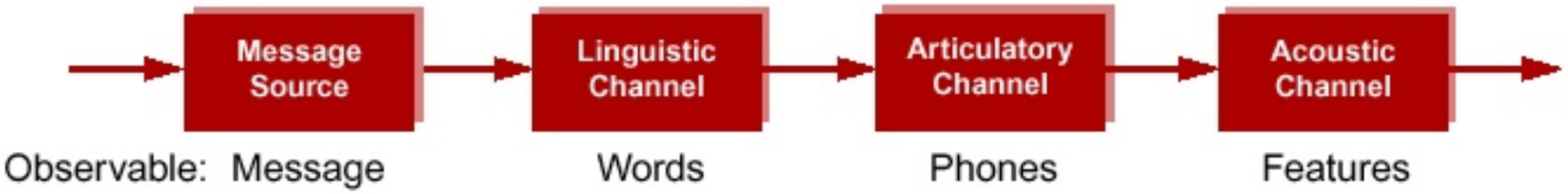
> X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

and from this source:

> F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, Boston, Massachusetts, USA, ISBN: 0-262-10066-5, 1998.

A NOISY COMMUNICATION CHANNEL MODEL OF SPEECH RECOGNITION

A noisy communication theory model for speech production and perception:



Bayesian formulation for speech recognition:

$$P(W|A) = P(A|W)P(W)/P(A)$$

Objective: minimize the word error rate by maximizing $P(W|A)$

Approach: maximize $P(A|W)$ (training)

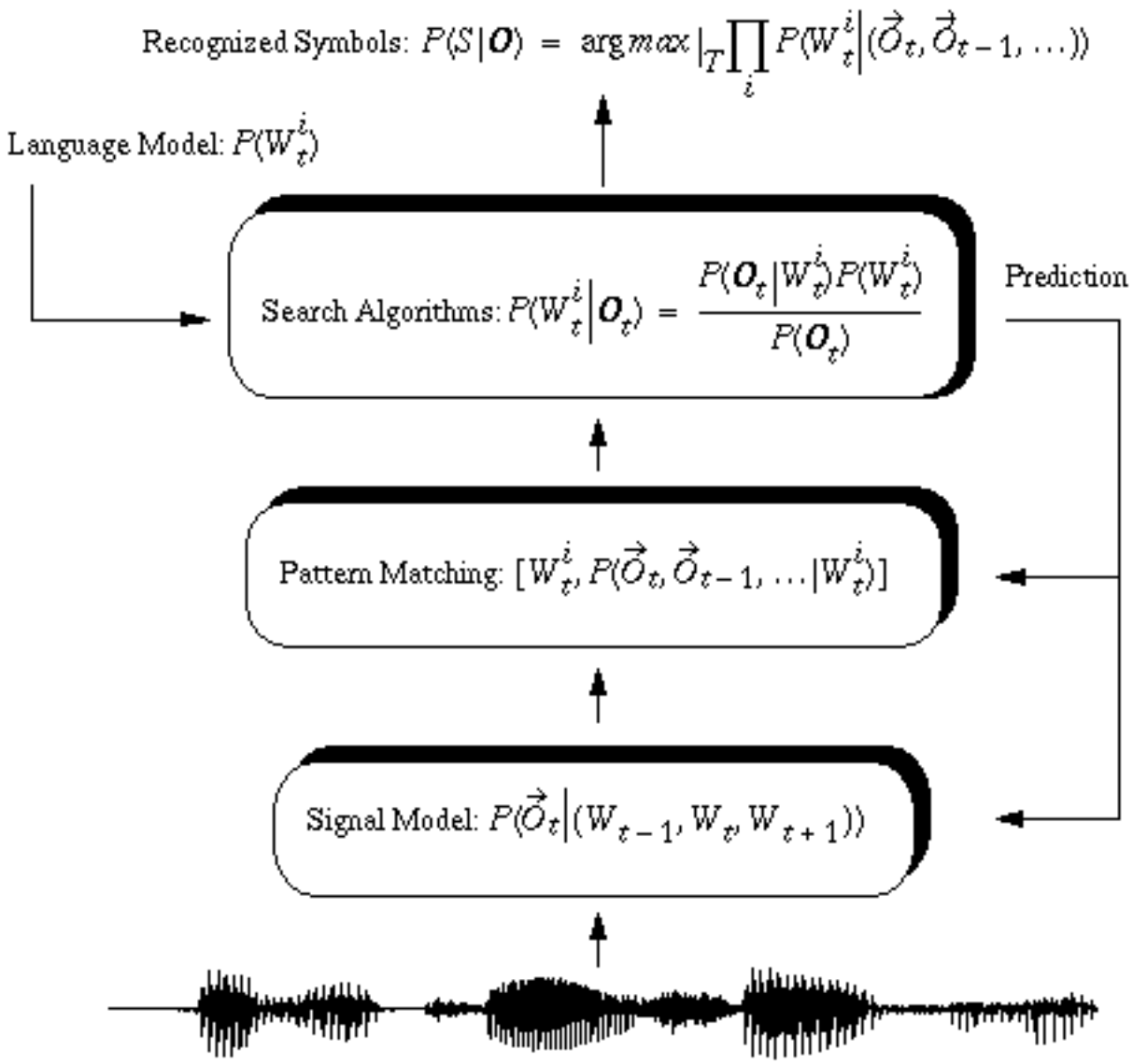
Components:

- $P(A|W)$: acoustic model (hidden Markov models, mixture of Gaussians)
- $P(W)$: language model (statistical, N-grams, finite state networks)
- $P(A)$: acoustics (ignore during maximization)

The language model typically predicts a small set of next words based on knowledge of a finite number of previous words (N-grams) — leads to search space reduction.

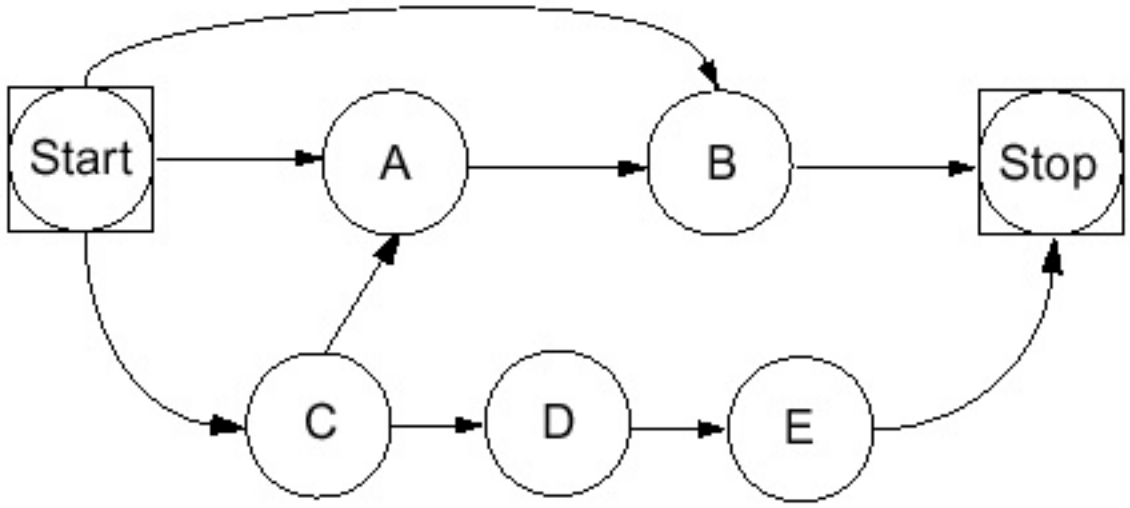
LANGUAGE MODELING IS SIMILAR TO PLAYING
... WHEEL ... OF ... FORTUNE

WORD PREDICTION IS CRITICAL



SYNTACTIC CONSTRAINTS CAN IMPROVE PERFORMANCE

- The search space for vocabularies of hundreds of words can become unmanageable if we allow any word to follow any other word (often called the no-grammar case)
- Our rudimentary knowledge of language tells us that, in reality, only a small subset of the vocabulary can follow a given word hypothesis, but that this subset is sensitive to the given word (we often refer to this as "context-sensitive")
- In real applications, user-interface design is crucial (much like the problem of designing GUI's), and normally results in a specification of a language or collection of sentence patterns that are permissible
- A simple way to express and manipulate this information in a dynamic programming framework is via a state machine:



For example, when you enter state C, you output one of the following words: {daddy, mommy}.

If:

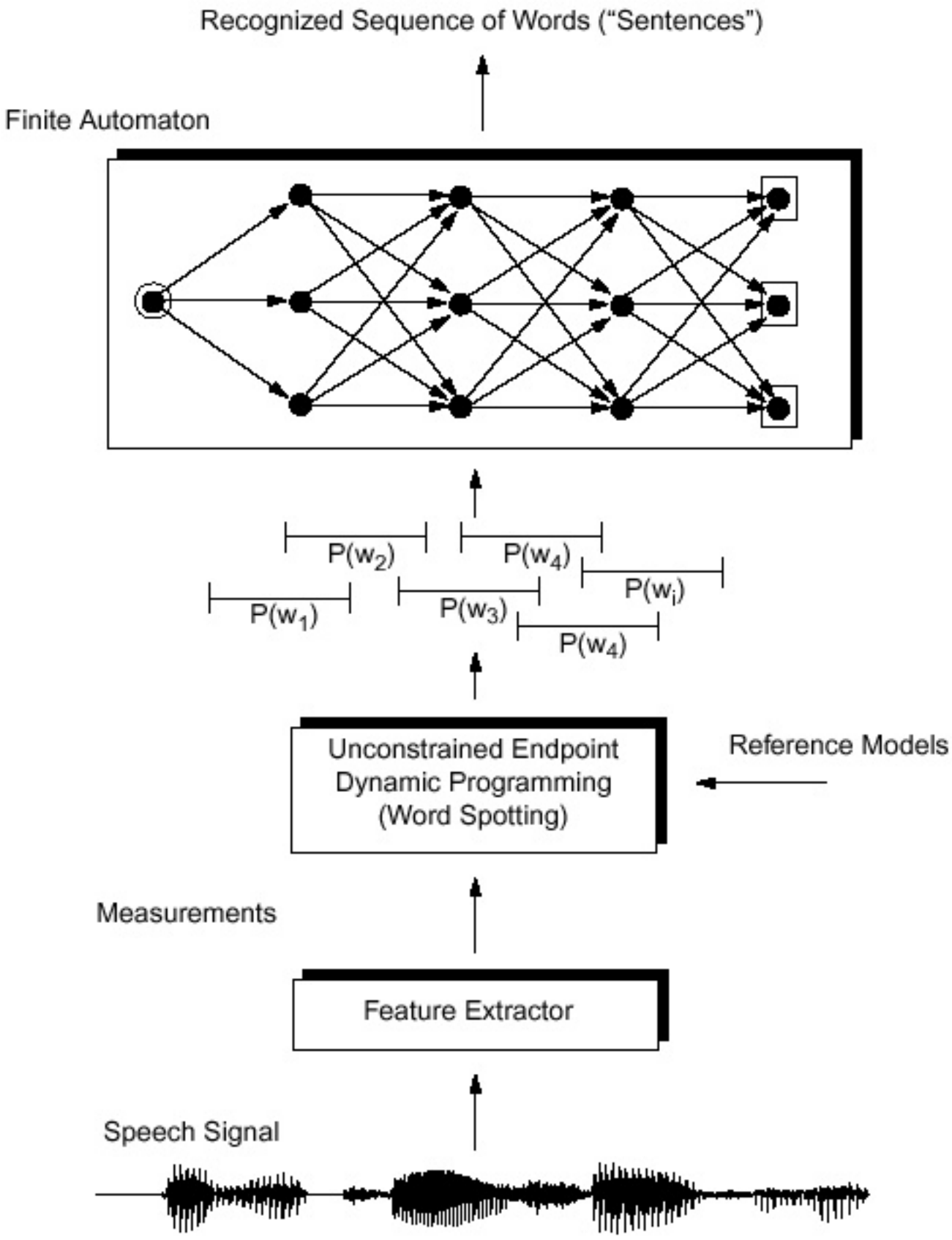
state A:	give
state B:	me
state C:	{daddy, mommy}
state D:	come
state E:	here

We can generate phrases such as:

Daddy give me

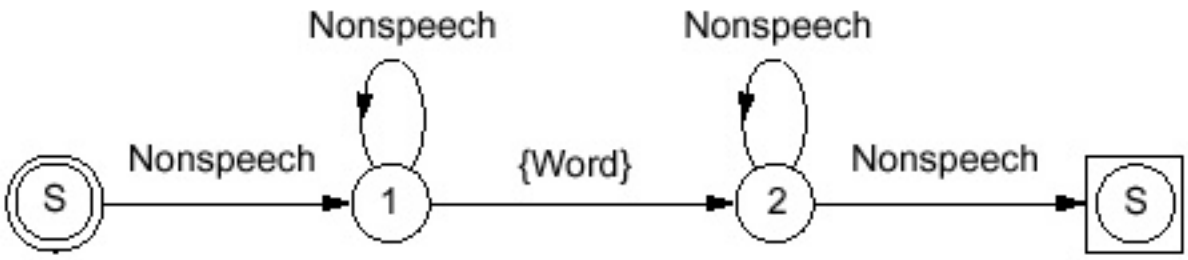
- We can represent such information numerous ways (as we shall see)

EARLY ATTEMPTS WERE AD HOC



NETWORK DECODING IS POPULAR FOR COMMAND AND CONTROL APPLICATIONS

Isolated Word Recognition:

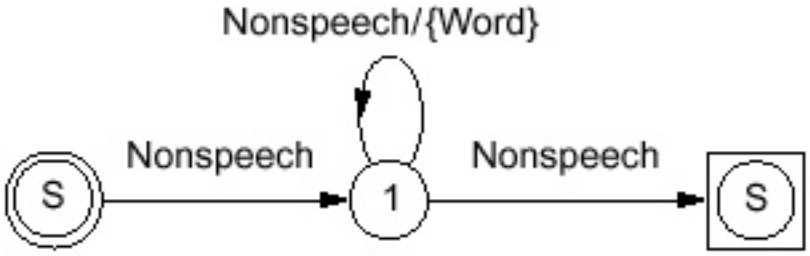


Nonspeech: typically an acoustic model of one frame in duration that models the background noise.

{Word}: any word from the set of possible words that can be spoken

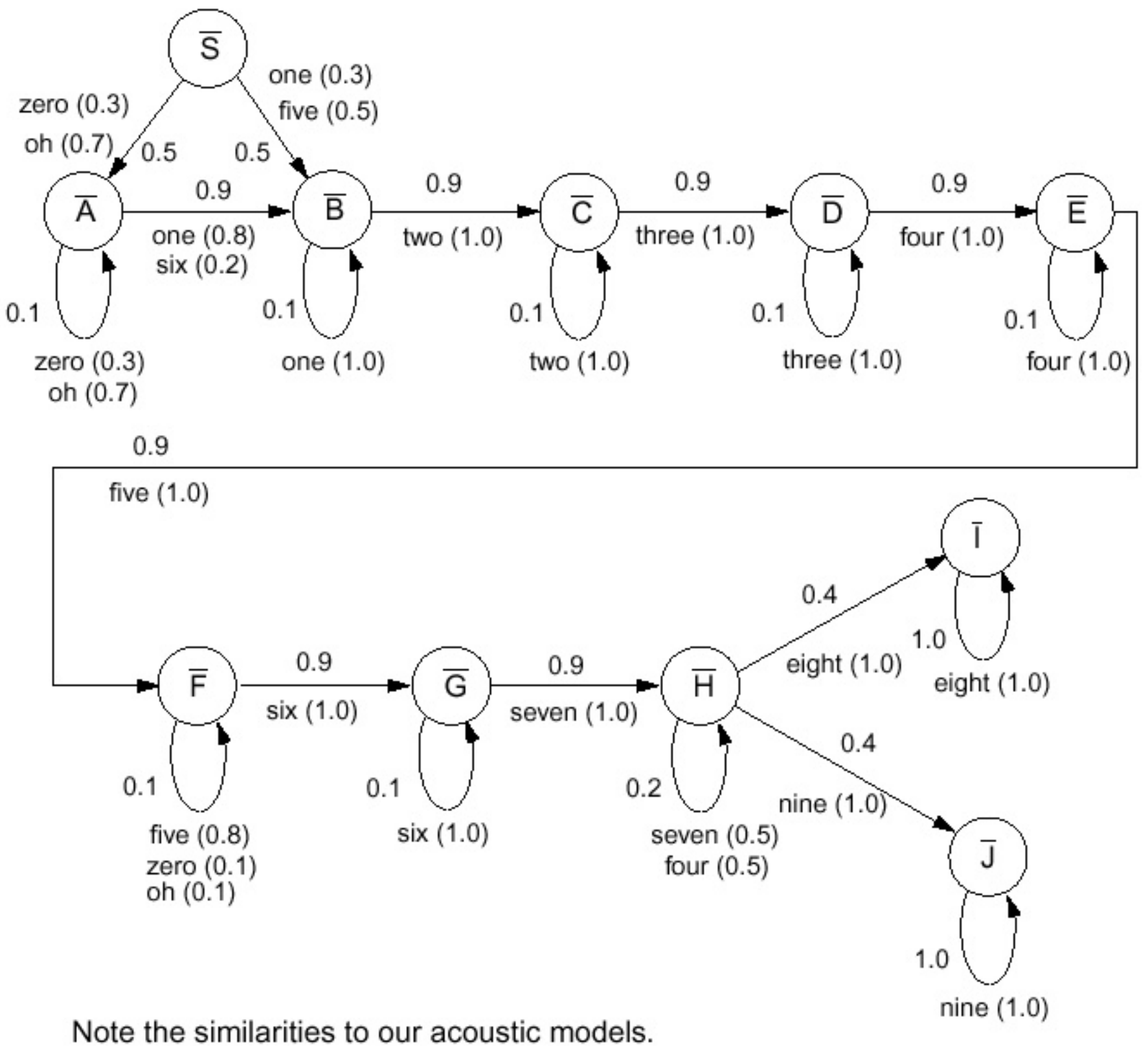
- The key point here is that, with such a system, the recognizer finds the optimal start/stop times of the utterance with respect to the acoustic model inventory (a hypothesis-directed search)

Simple Continuous Speech Recognition ("No Grammar"):



- system recognizes arbitrarily long sequences of words or nonspeech events

Consider the following state diagram showing a simple language model involving constrained digit sequences:



Note the similarities to our acoustic models.

What is the probability of the sequence “zero one two three four five zero six six seven seven eight eight” ?

How would you find the average length of a digit sequence generated from this language model?

In the terminology associated with formal language theory, this HMM is known as a *finite state automaton*.

The word *stochastic* can also be applied because the transitions and output symbols are governed by probability distributions.

Further, since there are multiple transitions and observations generated at any point in time (hence, ambiguous output), this particular graph is classified as a *nondeterministic* automaton.

In the future, we will refer to this system as a stochastic finite state automaton (FSA or SFSA) when it is used to more *linguistic* information.

We can also express this system as a *regular grammar*:

$\bar{S} \xrightarrow{p_1} \text{zero}, \vec{A}$ $\quad$ $\bar{D} \xrightarrow{p_{13}} \text{three}, \vec{D}$ $\quad$ $\bar{H} \xrightarrow{p_{25}} \text{eight}, \vec{I}$

$\bar{S} \xrightarrow{p_2} \text{oh}, \vec{A}$ $\quad$ $\bar{D} \xrightarrow{p_{14}} \text{four}, \vec{E}$ $\quad$ $\bar{H} \xrightarrow{p_{26}} \text{nine}, \vec{J}$

$\bar{S} \xrightarrow{p_3} \text{one}, \vec{A}$ $\quad$ $\bar{E} \xrightarrow{p_{15}} \text{four}, \vec{E}$ $\quad$ $\bar{I} \xrightarrow{p'_{27}} \text{eight}, \vec{I}$

$\bar{S} \xrightarrow{p_4} \text{five}, \vec{A}$ $\quad$ $\bar{E} \xrightarrow{p_{16}} \text{five}, \vec{F}$ $\quad$ $\bar{I} \xrightarrow{p''_{27}} \text{eight.}$

$\bar{A} \xrightarrow{p_5} \text{zero}, \vec{A}$ $\quad$ $\bar{F} \xrightarrow{p_{17}} \text{zero}, \vec{F}$ $\quad$ $\bar{I} \xrightarrow{p'_{28}} \text{nine}, \vec{J}$

$\bar{A} \xrightarrow{p_6} \text{oh}, \vec{A}$ $\quad$ $\bar{F} \xrightarrow{p_{18}} \text{oh}, \vec{F}$ $\quad$ $\bar{I} \xrightarrow{p''_{28}} \text{nine.}$

$\bar{A} \xrightarrow{p_7} \text{one}, \vec{B}$ $\quad$ $\bar{F} \xrightarrow{p_{19}} \text{five}, \vec{F}$

$\bar{A} \xrightarrow{p_8} \text{six}, \vec{B}$ $\quad$ $\bar{F} \xrightarrow{p_{20}} \text{six}, \vec{G}$

$\bar{B} \xrightarrow{p_9} \text{one}, \vec{B}$ $\quad$ $\bar{G} \xrightarrow{p_{21}} \text{six}, \vec{G}$

$\bar{B} \xrightarrow{p_{10}} \text{two}, \vec{C}$ $\quad$ $\bar{G} \xrightarrow{p_{22}} \text{seven}, \vec{H}$

$\bar{C} \xrightarrow{p_{11}} \text{two}, \vec{C}$ $\quad$ $\bar{H} \xrightarrow{p_{23}} \text{seven}, \vec{H}$

$\bar{C} \xrightarrow{p_{12}} \text{three}, \vec{D}$ $\quad$ $\bar{H} \xrightarrow{p_{24}} \text{four}, \vec{H}$

Note that rule probabilities are not quite the same as transition probabilities, since they need to combine transition probabilities and output probabilities. For example, consider p_7:

$$p_7 = (0.9)(0.8)$$

In general,

$$P(\underline{y} = y_k | \underline{x} = x_i) = \sum_j a_{ij} b(k)$$

Note that we must adjust probabilities at the terminal systems when the grammar is nondeterministic:

$$p_k = p'_k + p''_k$$

to allow generation of a final terminal.

Hence, our transition from HMMs to stochastic formal languages is clear and well-understood.

What types of language models are used?

- No Grammar (Digits)
- Sentence pattern grammars (Resource Management)
- Word Pair/Bigram (RM, Wall Street Journal)
- Word Class (WSJ, etc.)
- Trigram (WSJ, etc.)
- Back-Off Models (Merging Bigrams and Trigrams)
- Long Range N-Grams and Co-Occurrences (SWITCHBOARD)
- Triggers and Cache Models (WSJ)
- Link Grammars (SWITCHBOARD)

How do we deal with OOV and dysfluencies?

THE CHOMSKY HIERARCHY

We can categorize language models by their generative capacity:

Type of Grammar	Constraints	Automata
Phrase Structure	A -> B	Turing Machine (Unrestricted)
Context Sensitive	aAb -> aBb	Linear Bounded Automata (N-grams, Unification)
Context Free	A -> B Constraint: A is a non-terminal. Equivalent to: A -> w A -> BC where "w" is a terminal; B,C are non-terminals (Chomsky normal form)	Push down automata (JSGF, RTN, Chart Parsing)
Regular	A -> w A -> wB (Subset of CFG)	Finite-state automata (Network decoding)

- CFGs offer a good compromise between parsing efficiency and representational power.
- CFGs provide a natural bridge between speech recognition and natural language processing.