Deterministic Finite State Automata
Part 2

Adam Wyner
CS3518, Spring 2017
University of Aberdeen

Automata with Output

- We now have ways to formally define languages, and ways to automatically test whether a given string is a member of a language.
- We also have a simple model of a computer, which is done by adding output to string recognition.
- We extend DFSAs, so that instead of simply replying "yes" or "no" when presented with input, our abstract machine writes some output from an alphabet, determined by the input string.

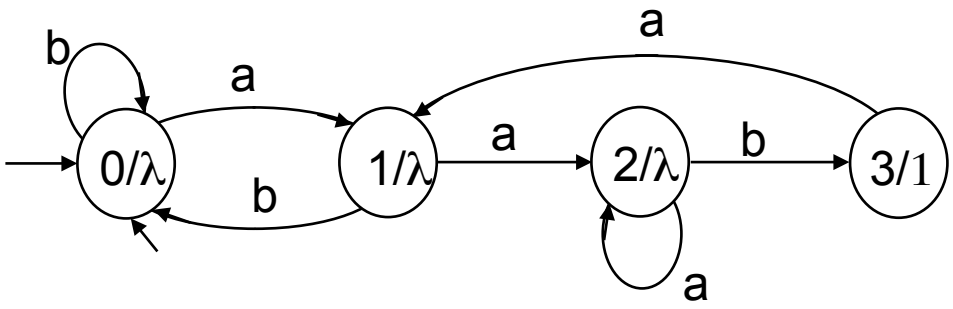
Moore Machines

- A Moore Machine is a 6-tuple (Q,I,T,E,Γ,O), where Q (states), I (initial), T (input alphabet), and E (transitions) are as for DFSA's. No final states F.
- Γ is a the output alphabet
- O is a function : Q $\rightarrow \Gamma$, defining the output corresponding to each state of the machine.
- $\lambda \in \Gamma$, where λ is null output.
- Graph: if (q, x) $\in$ O, then draw state q as:

- If state a is attained, output x.

Moore Machines

- Example: print out a 1 every time an aab substring is input.
- *aaababaaab* gives output of 11.
- Note that in state 0, we might have initial output but for λ.

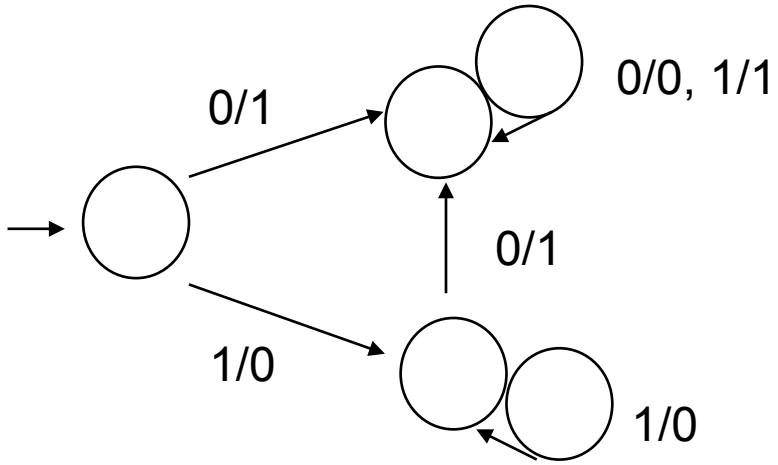


Mealy Machines

- A Mealy Machine is a 6-tuple (Q,I,T,E,Γ,O), where Q (states), I (initial), T (input alphabet), and E (transitions) are as for DFSA's. No final states F.
- Γ is the output alphabet.
- O is a function : Q x T $\rightarrow \Gamma$, defining the output corresponding to each state and symbol pair of the machine.
- Graph: if (q,t,x) $\in$ O, then draw it as an arc from q and labelled t/x.

Mealy Machines

- Example: read in a binary number and output a binary number that reverses the beginning sequence of numbers, but not the later sequence.
- 11101 input gives 00011 as output.



- Notice that there is only output on a transition.

Moore-Mealy Equivalence

- Let M be a Moore machine or a Mealy machine, with output alphabet Γ. Define $M^o(w)$ to be the output of M on input w.
- Let $M_1 = (Q_1, I_1, T_1, E_1, \Gamma_1, O_1)$ be a Moore machine
- Let $M_2 = (Q_2, I_2, T_2, E_2, \Gamma_2, O_2)$ be a Mealy machine.
- M_1 and M_2 are equivalent if $T_1 = T_2$, where $M^o(\lambda) = b$, and for strings $w \in T_1^*$, $M_1^0(w) = b\, M_2^0(w)$.
- Theorem: Moore-Mealy Equivalence

  If M1 is a Moore machine, then there exists a Mealy machine, M2, equivalent to M1.

  If M2 is a Mealy machine, then there exists a Moore machine, M1, equivalent to M2.

Moore-Mealy Comparison

Moore

- Output depends only on the present state
- Generally, has more states than a Mealy Machine.
- Input directly changes output with respect to a state.

Mealy

- Output depends on current state and current input.
- Generally, has fewer states than a Moore Machine.
- Output changes as states change (can be associated with clock ticks).

Varieties of FSAs

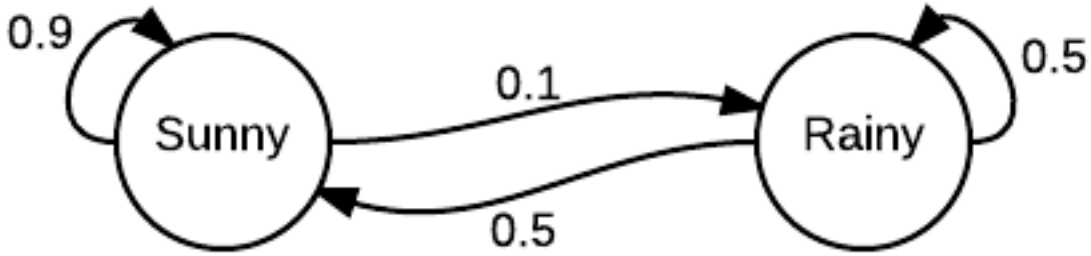
- FSAs with output have applications e.g. in the design of machines (e.g. a vending machine: reads pounds, pennies, etc; outputs a chocolate bar); coding and decoding of messages; etc.

- Many other variations on the theme of FSAs exist:
  - Markov models (FSAs whose edges have probabilities)
  - These have many applications, e.g. in Natural Language Processing

Markov Models

- Markov Models model the probability of one state following another. (Assumption: only the previous state matters.)

- Example: a simple weather model:

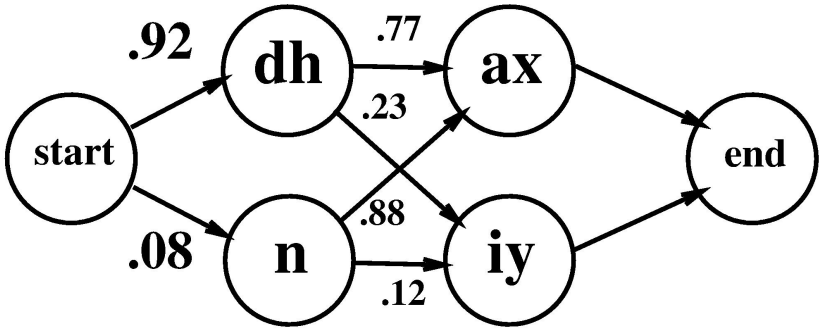


- The probability that is it rainy after it has been sunny is .1%
- The probability that it is sunny after it has been sunny is .9%

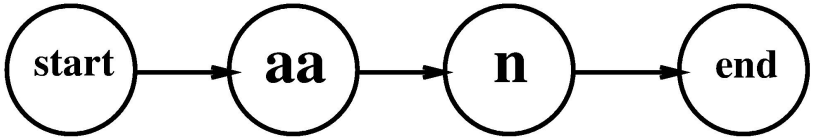
Applications of Markov Models

- In Natural Language Processing, Markov Models are often used.
- For instance, they can model how a word is pronounced as phones ("speech sounds").
- State transition model, where each state is a phone. A path through the states is the pronunciation of a word or sequence of words.
- Assign transitions a probability (not an output).
- A path gives the likelihood of a sequence of phones, i.e., the pronunciation of a word or sequence of words.
- Probability of a path through the model is the product of the probabilities of the transitions.

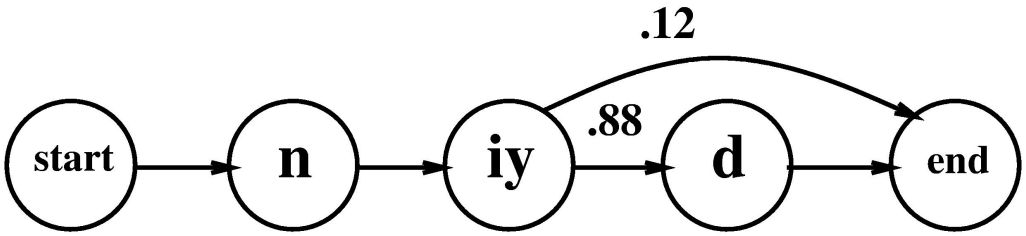
Markov Models for Pronunciation



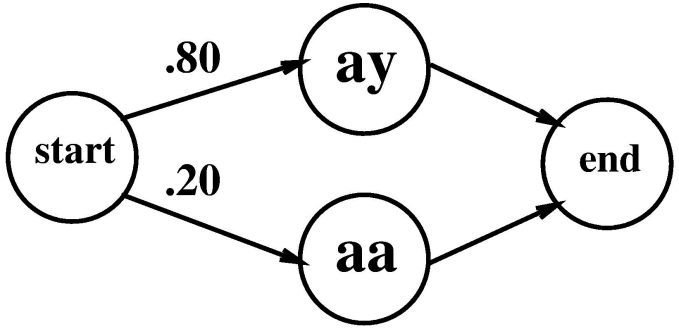
Word model for "the"



Word model for "on"



Word model for "need"



Word model for "I"

Pronunciation Example

- Suppose this string of phones has been recognized:

  aa n iy dh ax

- Three different utterances (i.e., stings of words) may explain what was recognized:

  1. I/[aa] need/[n iy] the/[dh ax]
  2. I/[aa] the/[n iy] the/[dh ax]
  3. On/[aa n] he/[iy] the/[dh ax]

Probabilities Based on the Markov Model

- Probabilities of paths:

- I/[aa] need/[n iy] the/[dh ax]

  .2 * .12 * .92*.77 = .017

- I/[aa] the/[n iy] the/[dh ax]

  .2 * .08*.12 * .92*.77 = .001

- On/[aa n] he/[iy] the/[dh ax]

  1 * .1 * .92*.77 = .071

Ranked Probabilities

- Ranked list of paths:
  - On he the (.071)
  - I need the (.017)
  - I the the (.001)
  - etc..
- System would only keep top N

Pronunciation Example

- These probabilities can help a speech recognition system
- Other information is needed as well:
  - Probability of each phone
  - Probability of each word
  - Probability that one word follows another
    (Can “on he the” be part of an English sentence?)

Summing Up

- Finite automata are flexible tools.
  - Important in practical applications.
- DFSAs are relatively simple automata.
  - Variants of DFSAs can produce output (and/or use probabilities).
- Later in this course: More complex automata called Turing Machines (TM).
  - Some variants of TMs produce output.