

Regular Expressions and Finite State Automata

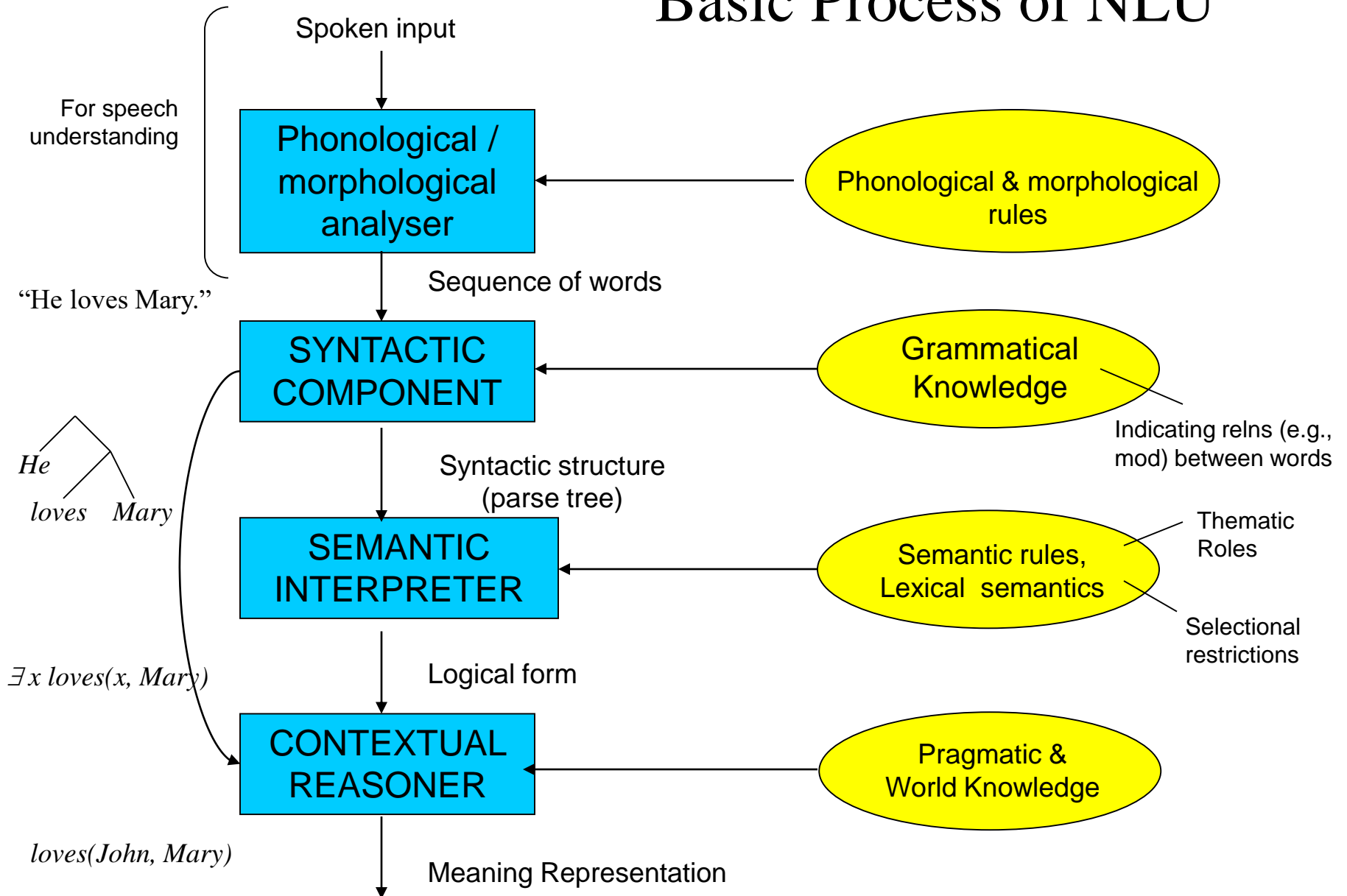
Lecture #2

**SNU 4th Industrial Revolution Academy:
Artificial Intelligence Agent**

Natural Language Understanding

- Associate each input (acoustic signal/character string) with a meaning representation.
- Carried out by a series of components:
 - Each component acts as a translator from one representation to another
 - In general, each component adds successively 'richer' information to the output

Basic Process of NLU



Representations and Algorithms for NLP

- Representations: formal models used to capture linguistic knowledge
- Algorithms manipulate representations to analyze or generate linguistic phenomena

NLP Representations

- State Machines
 - FSAs, FSTs, HMMs, ATNs, RTNs
- Rule Systems
 - CFGs, Unification Grammars, Probabilistic CFGs
- Logic-based Formalisms
 - 1st Order Predicate Calculus, Temporal and other Higher Order Logics
- Models of Uncertainty
 - Bayesian Probability Theory

NLP Algorithms

- Most are parsers or transducers: accept or reject input, and construct new structure from input
 - State space search
 - Pair a partial structure with a part of the input
 - Spaces too big and 'best' is hard to define
 - Dynamic programming
 - Avoid recomputing structures that are common to multiple solutions

Regular Expressions

- Can be viewed as a way to specify:
 - Search patterns over text string
 - Design of a particular kind of machine, called a Finite State Automaton (FSA)
- These are really equivalent

Uses of Regular Expressions in NLP

- As grep, perl: Simple but powerful tools for large corpus analysis and ‘shallow’ processing
 - What word is most likely to begin a sentence?
 - What word is most likely to begin a question?
 - In your own email, are you more or less polite than the people you correspond with?
- With other unix tools, allow us to
 - Obtain word frequency and co-occurrence statistics
 - Build simple interactive applications (e.g. Eliza)
- Regular expressions define regular languages or sets

Regular Expressions

- **Regular Expression:** Formula in algebraic notation for specifying a set of strings
- **String:** Any sequence of alphanumeric characters
 - Letters, numbers, spaces, tabs, punctuation marks
- **Regular Expression Search**
 - **Pattern:** specifying the set of strings we want to search for
 - **Corpus:** the texts we want to search through

Some Examples

RE	Description	Uses
/./	Wild card; Any char	A blank line?
/a/	Any 'a'	Line with words?
/[ab]/	A choice	Rhyming words?
/[a-z]/	l.c. char (range)	Common noun?
/[A-Z]/	u.c. char	Proper noun?
/[^?!.]/	Neg of set	Not S-final punc

RE	Description	Uses?
/a*/	Zero or more a's	Optional doubled modifiers (words)
/a+/	One or more a's	Non-optional...
/a?/	Zero or one a's	Optional...
/cat dog/	'cat' or 'dog'	Words modifying pets
/^cat\.\$/	A line that contains only cat. ^anchors beginning, \$ anchors end of line.	??
/bun\B/	Beginnings of longer strings	Words prefixed by 'un'

RE	E.G.
/pupp(y ies)/	Morphological variants of 'puppy'
/ (.+)ier and \1ier /	happier and happier, fuzzier and fuzzier

Optionality and Repetition

- `/[Ww]oodchucks?/` matches woodchucks, Woodchucks, woodchuck, Woodchuck
- `/colou?r/` matches color or colour
- `/he{3}/` matches heee
- `/(he){3}/` matches hehehe
- `/(he){3,}/` matches a sequence of at least 3 he's

Operator Precedence Hierarchy

- | | |
|------------------------|---------------|
| 1. Parentheses | () |
| 2. Counters | * + ? {} |
| 3. Sequence of Anchors | the ^my end\$ |
| 4. Disjunction | |

Examples

/moo+/

/try|ies/

/and|or/

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

/the/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

/the/

/[tT]he/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

/the/

/[tT]he/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

/the/

/[tT]he/

/\b[tT]he\b/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner “the”:

/the/

/[tT]he/

/\b[tT]he\b/

/(^|^[^a-zA-Z])[tT]he[^a-zA-Z]/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

The Two Kinds of Errors

- The process we just went through was based on fixing errors in the regular expression
 - Errors where some of the instances were missed (judged to not be instances when they should have been) – False negatives
 - Errors where the instances were included (when they should not have been) – False positives
- This is pretty much going to be the story of the rest of the course!

Substitutions (Transductions)

- Sed or 's' operator in Perl
 - s/regexp1/pattern/
 - s/I am feeling (.++)/You are feeling \1?/
 - s/I gave (.+) to (.+)/Why would you give \2 \1?/

Examples

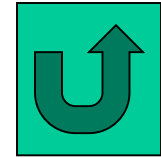
- Predictions from a news corpus:
 - Which candidate for Governor of California is mentioned most often in the news? Is going to win?
 - What stock should you buy?
 - Which White House advisers have the most power?
- Language use:
 - Which form of comparative is more frequent: ‘oftener’ or ‘more often’?
 - Which pronouns are conjoined most often?
 - How often do sentences end with infinitival ‘to’?
 - What words most often begin and end sentences?
 - What’s the most common word in your email? Is it different from your neighbor?

- Personality profiling:
 - Are you more or less polite than the people you correspond with?
 - With labeled data, which words signal friendly messages vs. unfriendly ones?

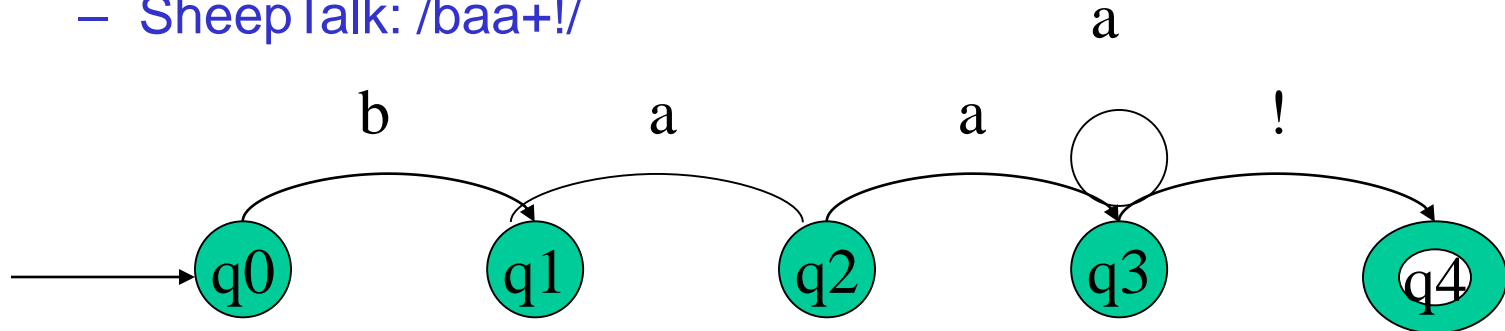
Finite State Automata

- Regular Expressions (REs) can be viewed as a way to describe machines called Finite State Automata (FSA, also known as automata, finite automata).
- FSAs and their close variants are a theoretical foundation of much of the field of NLP.

Finite State Automata



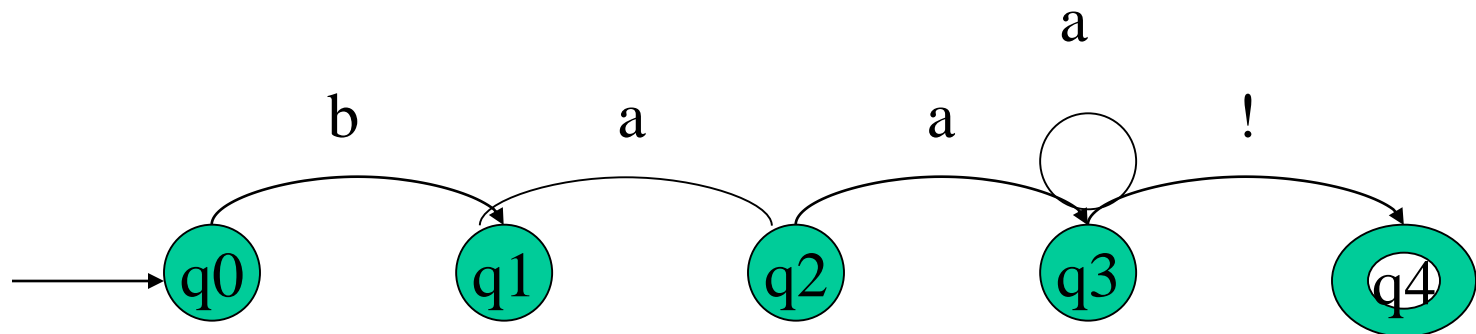
- FSAs recognize the regular languages represented by regular expressions
 - SheepTalk: /baa+!/



- Directed graph with labeled nodes and arc transitions**
- Five states:** q0 the start state, q4 the final state, 5 transitions

Formally

- FSA is a 5-tuple consisting of
 - Q : set of states $\{q_0, q_1, q_2, q_3, q_4\}$
 - Σ : an alphabet of symbols $\{a, b, !\}$
 - q_0 : a start state
 - F : a set of final states in Q $\{q_4\}$
 - $\delta(q, i)$: a transition function mapping $Q \times \Sigma$ to Q



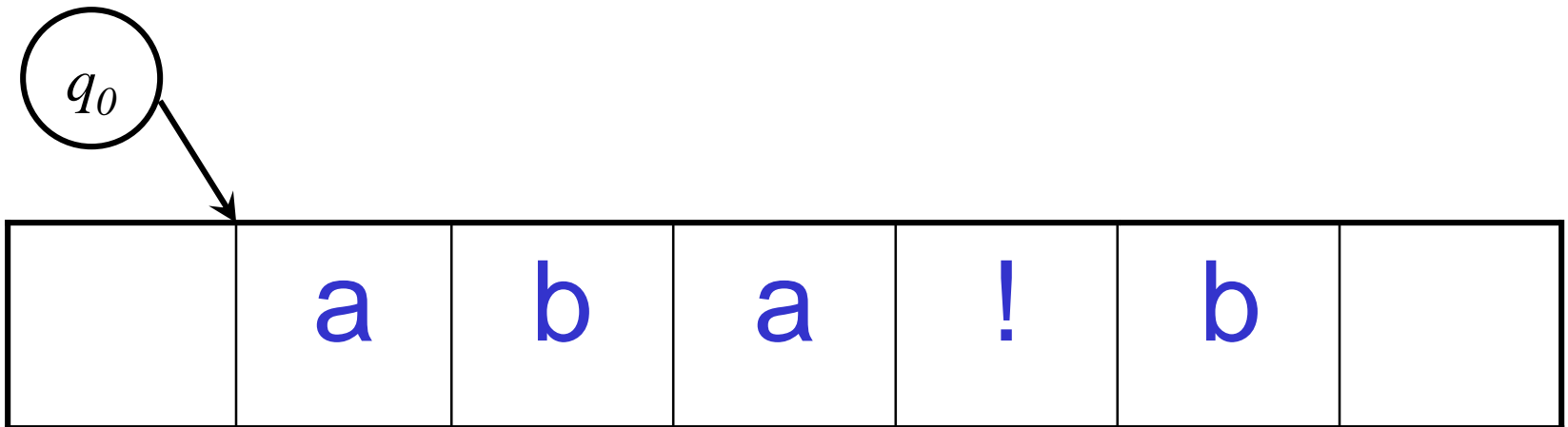
State Transition Table for SheepTalk

State	Input		
	b	a	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

Recognition

- Recognition (or acceptance) is the process of determining whether or not a given input should be accepted by a given machine.
- In terms of REs, it's the process of determining whether or not a given input matches a particular regular expression.
- Traditionally, recognition is viewed as processing an input written on a tape consisting of cells containing elements from the alphabet.

- FSA recognizes (**accepts**) strings of a regular language
 - baa!
 - baaa!
 - baaa!
 - ...
- Tape metaphor: a rejected input



D-Recognize

Function D-Recognize(tape, machine) **returns** accept or reject

Index \leftarrow Beginning of tape

Current-state \leftarrow Initial state of the machine

loop

If end of input has been reached **then**

If current-state is an accept state **then**

return accept

Else

return reject

elseif transition-table[current-state,tape[index]] is empty **then**

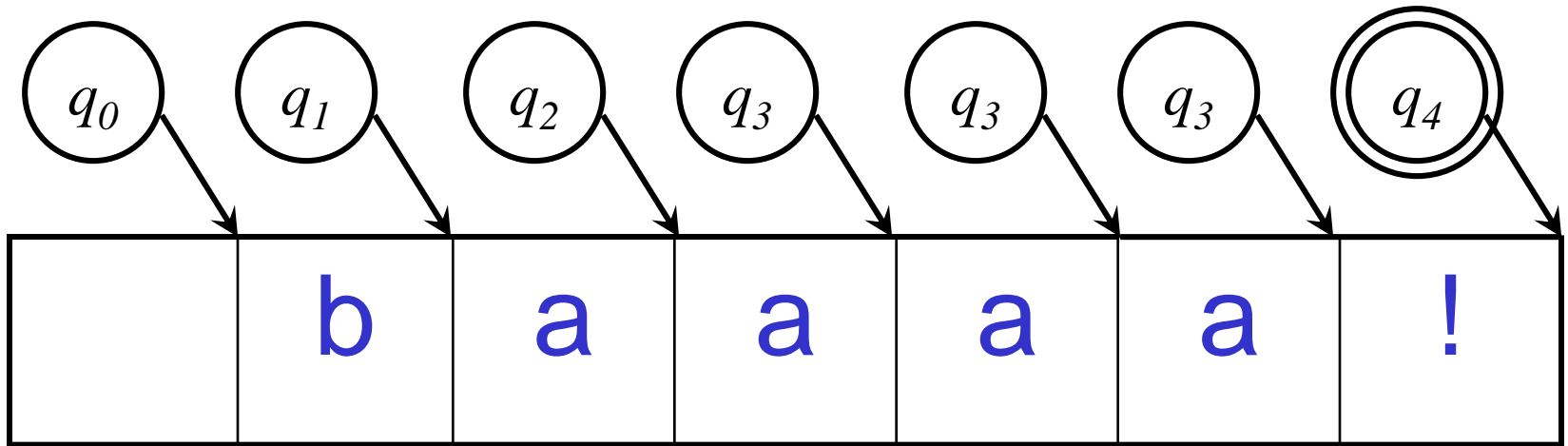
return reject

else

Current-state \leftarrow transition-table[current-state,tape[index]]

Index \leftarrow index + 1

end



State	Input		
	b	a	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

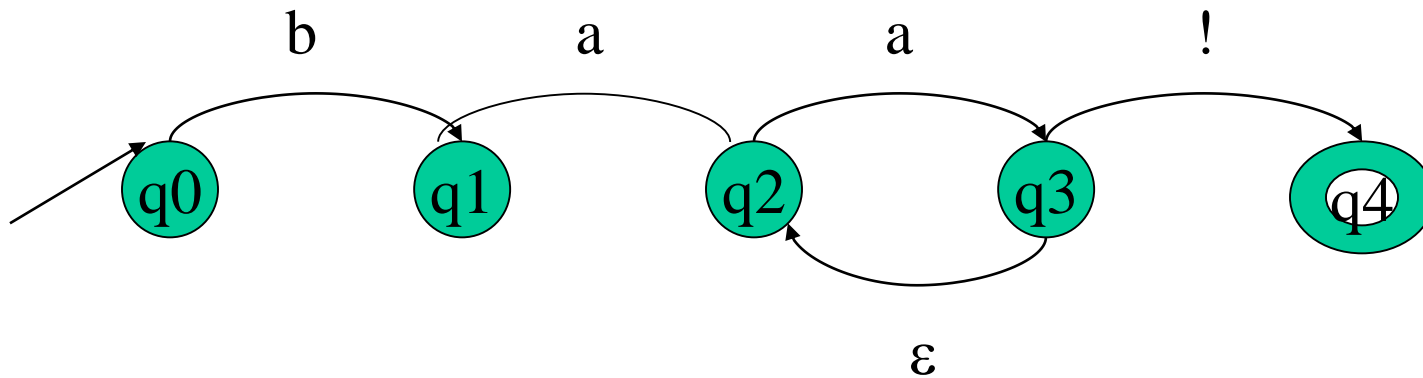
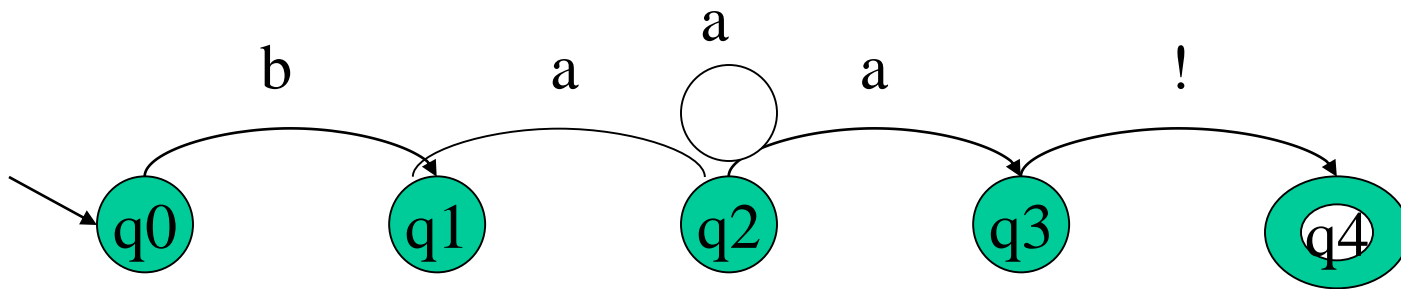
Key Points About D-Recognize

- Deterministic means that the code always knows what to do at each point in the process
- Recognition code is universal for all FSAs. To change to a new formal language:
 - change the alphabet
 - change the transition table
- Searching for a string using a RE involves compiling the RE into a table and passing the table to the interpreter

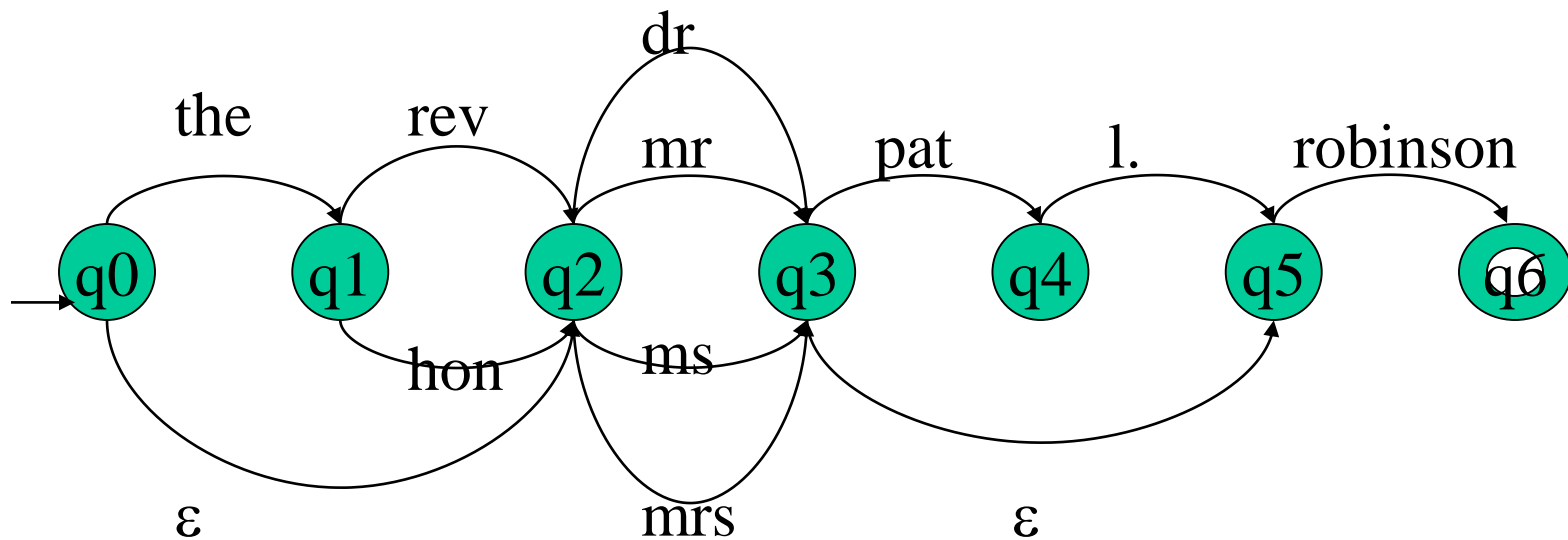
Determinism and Non-Determinism

- **Deterministic:** There is at most one transition that can be taken given a current state and input symbol.
- **Non-deterministic:** There is a choice of several transitions that can be taken given a current state and input symbol. (The machine doesn't specify how to make the choice.)

Non-Deterministic FSAs for SheepTalk



FSAs as Grammars for Natural Language



Can you use a regexpr to capture this too?

Problems of Non-Determinism

- ‘Natural’....but at any choice point, we may follow the wrong arc
- Potential solutions:
 - Save backup states at each choice point
 - Look-ahead in the input before making choice
 - Pursue alternatives in parallel
 - Determinize our NFSA's (and then minimize)
- FSAs can be useful tools for recognizing – and generating – subsets of natural language
 - But they cannot represent all NL phenomena (Center Embedding: The mouse the cat ... chased died.)

Recognition as Search

- Systematically Searching for Solutions → state-space search algorithm
- “Order” – in which the states in the space are considered
- Depth-first search or Last in First Out (LIFO) –stack
 - Figure 2.22
- Breadth-first search or First in First Out (FIFO) – queue
 - Figure 2.23

Regular Expressions as NFAs

- Regular expressions can easily be represented using NFAs
- We can group regular expressions into 4 different components

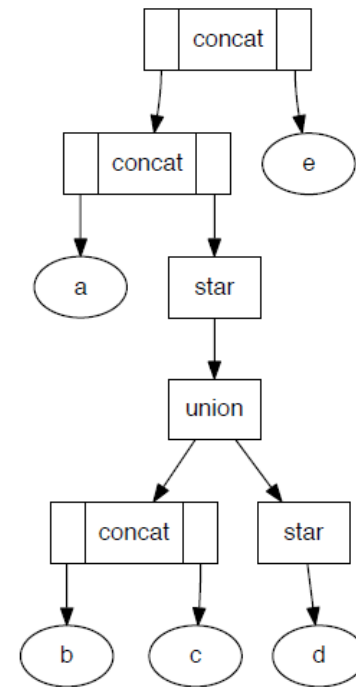
Character a single character:
/a/

Concatenation two adjacent
expressions: /ab/

Union two OR'd
expressions: /a|b/

Kleene star zero or more
repetitions: /a*/

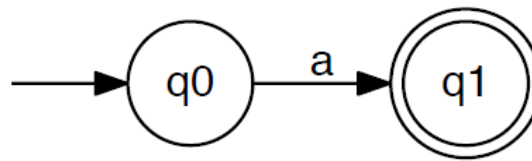
/a(bc|d*)*e/ can be
viewed as



RE to NFA

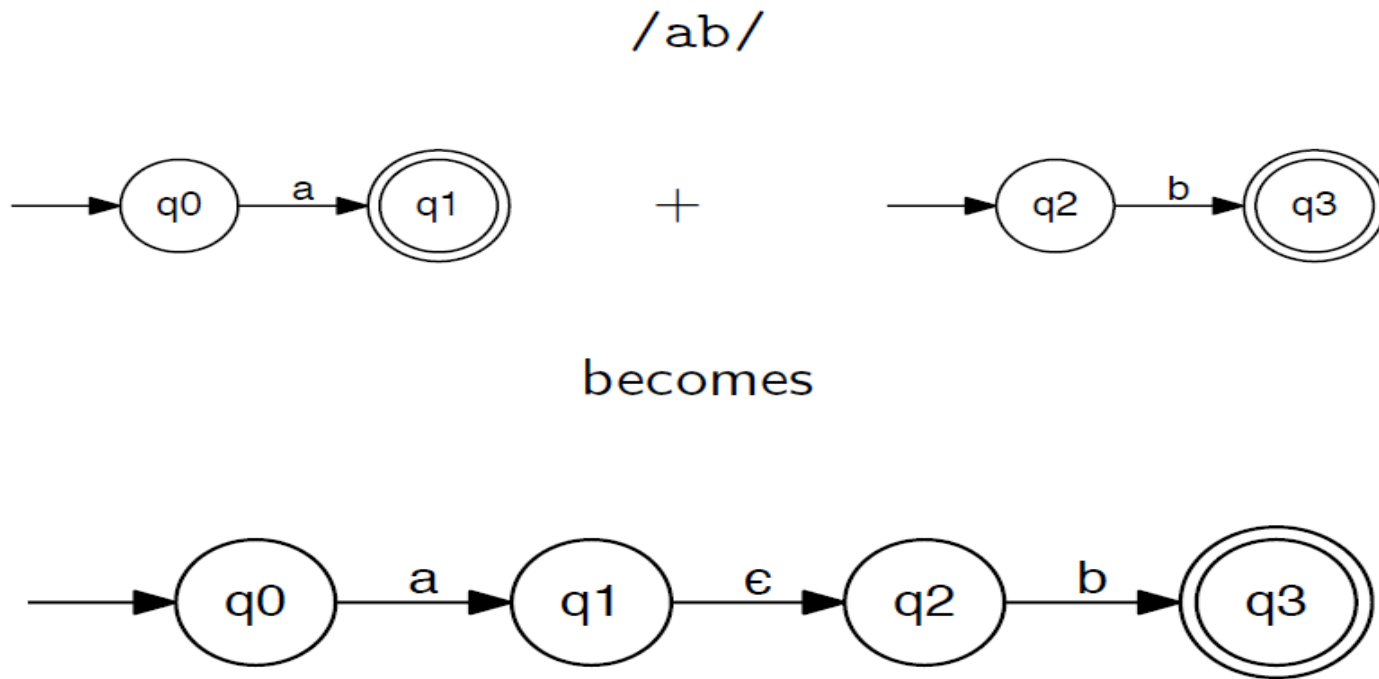
- Character

/a/



RE to NFA

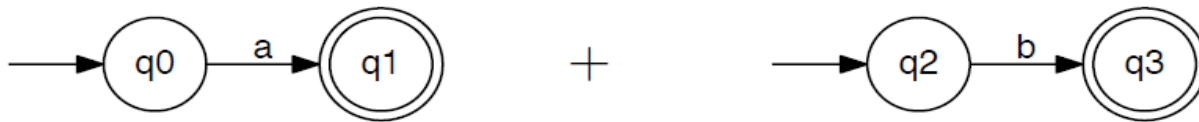
- Concatenation



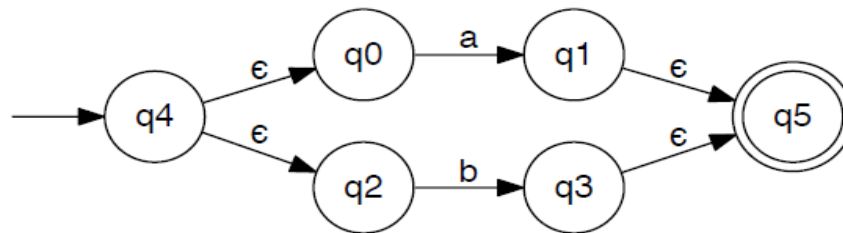
RE to NFA

- Union

$/a|b/$



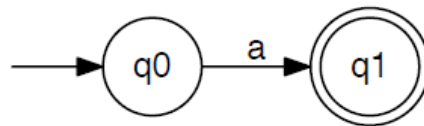
becomes



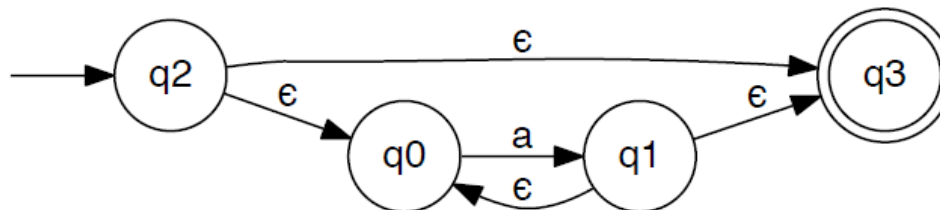
RE to NFA

- Kleene Star

$/a^*/$



becomes



RE to NFA

- An Example Conversion

$/a(bc|d^*)^*e/$

becomes

