LANGUAGE, FSA & REGEX

NATURAL LANGUAGE PROCESSING - CS 322.00

Blake Howald

September 13, 2017

AGENDA

- Presentation Sign-Up (circulating), Grading, …etc.
- Questions?
- Natural, Formal and Regular Languages
- Finite State Automata

NATURAL LANGUAGE

- Characteristics that define Natural Languages
  - Productivity/ Creativity
  - Discreteness
  - Recursion
  - Arbitrariness
  - Displacement
  - Cultural Transmission / Variability
  - Interchangeability / Semanticity
  - Context / Feedback

(Hockett 1960)

NATURAL LANGUAGE

- Productivity/ Creativity
  - Pascale and Yeardley brought the tent into the family room
- Discreteness
  - Pascale and Yeardley [***f***]ought [***a dragon***] [***with***] the family [***b***]room
- Recursion
  - Pascale and Yeardley, [***who are sisters***], fought a dragon with the family broom
- Arbitrariness
  - [***Pascale et Yeardley, qui sont des sœurs, ont combattu un dragon avec le balai familial***] (Google Translate)

NATURAL LANGUAGE

- Cultural Transmission / Variability
  - Pascale and Yeardley, who are sisters, fought a dragon with the family [***push***|***straw***|***house***|***brush***|***shop***|***corn***] broom. (Dictionary of American Regional English)
- Displacement
  - Pascale and Yeardley, who are sisters, fought a dragon with the family broom, [***and probably always will***].
- Interchangeability / Semanticity
  - What other things can Pascale and Yeardley use to fight the dragon?

NATURAL LANGUAGE

- Context / Feedback
  - Pascale and Yeardley, who are sisters, fought a dragon with the family broom, and probably always will.
  - Do I mean?

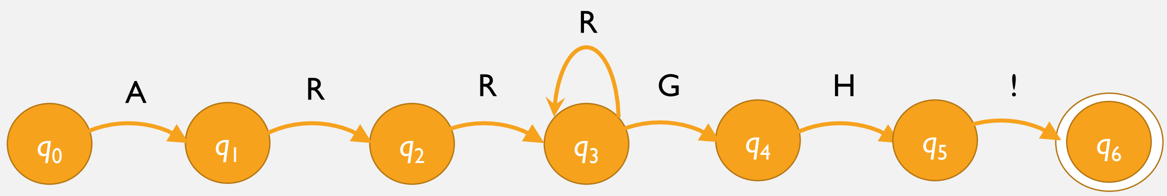
FORMAL LANGUAGE

- A ***Formal Language*** can be used to model different aspects of a ***Natural Language*** (phonetics, phonology, morphology, syntax).
- A Formal Language can be defined by a model that both accepts and generates all and only the strings of the formal language.
  - Alphabet: $\Sigma = \{\}$
  - Model: m
  - Formal Language: L(m)

REGULAR LANGUAGE (1^{ST} PASS)

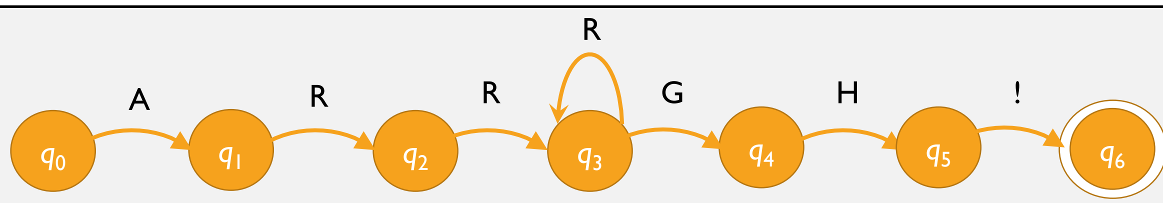
- A ***Regular Language*** is a type of Formal Language where m is captured by a ***Finite State Automata*** (FSA), requiring five parameters:
  - Finite Set of N States: $Q=q_0q_1q_2\ldots q_{N-1}$
  - Finite Alphabet: $\Sigma = \{\}$
  - Start State: q_0
  - Set of final states: $F \subseteq Q$
  - Transition Function: $\delta(q,i)$
    - Given $q \in Q$ and $i \in \Sigma$, $\delta(q,i)$ returns $q' \in Q$

DFSA REPRESENTATION



Finite Set of N States: $Q=\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$
Finite Alphabet: Σ = {A, R, G, H, !}
Start State: q_0
Set of final (or accepting) states: $F \subseteq Q = \{q_6\}$

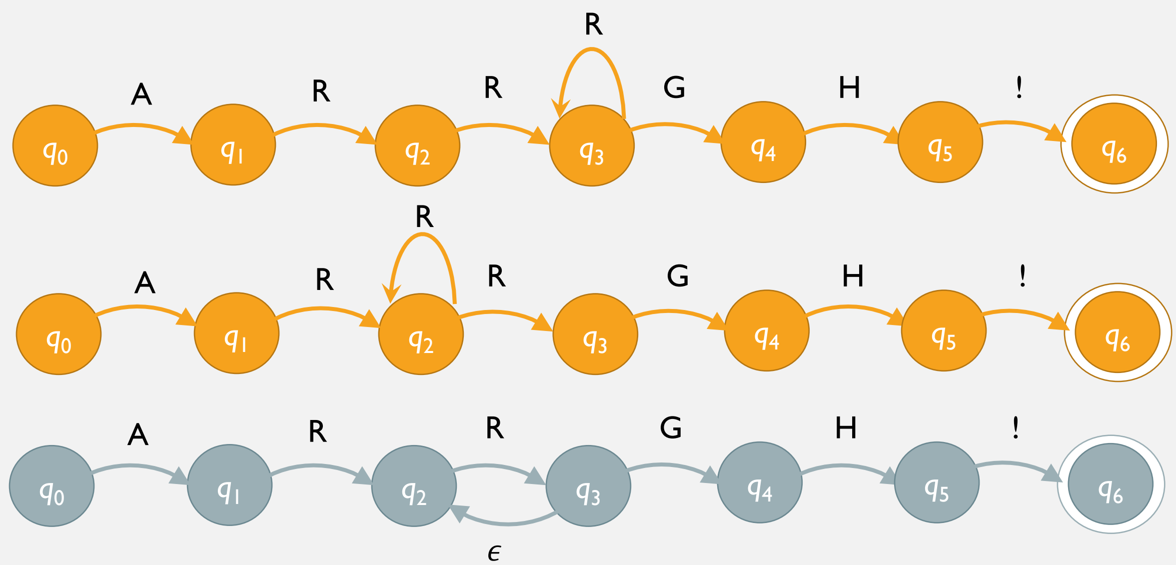
International Talk Like a Pirate Day
September 19, 2017

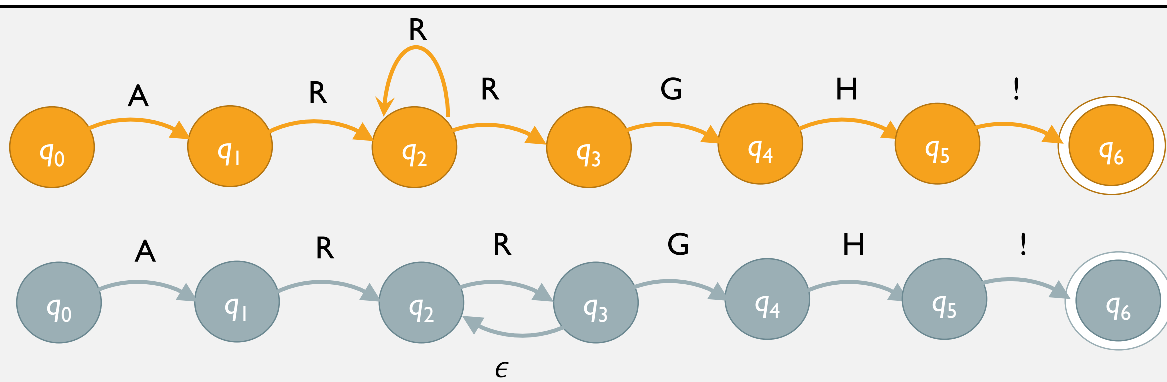


Transition Function: $\delta(q,i)$
Given $q \in Q$ and $i \in \Sigma$, $\delta(q,i)$ returns $q' \in Q$

	Input				
State	A	R	G	H	!
0	1	∅	∅	∅	∅
1	∅	2	∅	∅	∅
2	∅	3	∅	∅	∅
3	∅	3	4	∅	∅
4	∅	∅	∅	5	∅
5	∅	∅	∅	∅	6
6:	∅	∅	∅	∅	∅

DFSA VS NFSA





	Input				
State	A	R	G	H	!
0	1	∅	∅	∅	∅
1	∅	2	∅	∅	∅
2	∅	2, 3	∅	∅	∅
3	∅	∅	4	∅	∅
4	∅	∅	∅	5	∅
5	∅	∅	∅	∅	6
6:	∅	∅	∅	∅	∅

	Input					
State	A	R	G	H	!	ϵ
0	1	∅	∅	∅	∅	∅
1	∅	2	∅	∅	∅	∅
2	∅	3	∅	∅	∅	∅
3	∅	∅	4	∅	∅	2
4	∅	∅	∅	5	∅	∅
5	∅	∅	∅	∅	6	∅
6:	∅	∅	∅	∅	∅	∅

NFSA ACCEPTANCE STRATEGIES

- We don't care (yet) about how decisions are made, but accepting a string with an NFSA (algorithmically), can be accomplished with one of three strategies:
  - ***Backup*** – place a "marker" at a choice point and, if failure, start again at the marker and exhaust all possibilities.
  - ***Look-Ahead*** – look ahead in the input (based on some window), will revisit with parsing
  - ***Parallelism*** – run multiple paths in parallel

NFSA ACCEPTANCE STRATEGIES

- The "Back-Up" Strategy relies on approaches to search:
  - ***Depth-First (LIFO)***
    - *Stack* data structure
    - Good if memory size is small to medium
    - Standard concern with infinite loops
  - ***Breadth-First (FIFO)***
    - *Queue* data structure
    - Bad if memory size is large
    - Similar concerns with infinite loops

D|NFSA FINAL NOTES

- All NFSAs can be converted to an equivalent DFSA
  - Larger number of states – a DFSA is built for each possible path in the NFSA (***parallelism*** strategy) – up to 2^N states in the resulting DFSA
- Any FSA can be described by a Regular Expression (REGEX)
  - And any REGEX can implement a FSA
- More on REGEXs Friday (9/15/17)

REGULAR LANGUAGE (1ST PASS)

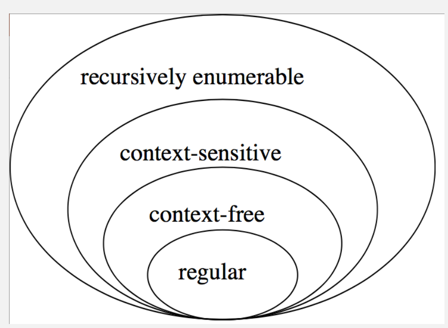
- A ***Regular Language*** is a type of Formal Language where *m* is captured by a ***Finite State Automata*** (FSA), requiring five parameters:
  - Finite Set of N States: $Q=q_0q_1q_2\ldots q_{N-1}$
  - Finite Alphabet: $\Sigma = \{\}$
  - Start State: q_0
  - Set of final states: $F \subseteq Q$
  - Transition Function: $\delta(q,i)$
    - Given $q \in Q$ and $i \in \Sigma$, $\delta(q,i)$ returns $q' \in Q$

REGULAR LANGUAGE (2^{ND} PASS)

- A ***Regular Language*** ∅ is a type of Formal Language (subset) where *m* is captured by a ***Finite State Automata*** (FSA), requiring five parameters:
  - Finite Set of N States: $Q = q_0 q_1 q_2 \ldots q_{N-1}$
  - Finite Alphabet: $\Sigma = \{\} \cup \epsilon$
  - Start State: q_0
  - Set of final states: $F \subseteq Q$
  - Transition Function: $\delta(q,i)$
    - Given $q \in Q$ and $i \in \Sigma$, $\delta(q,i)$ returns $q' \in Q$

REGULAR LANGUAGES (LAST PASS, FOR NOW)

- Regular Languages (Type-3 Grammar)
  - Recognition with FSA
  - Is a subset of:
    - Context-Free (Type-2 Grammar)
    - Context-Sensitive (Type-1 Grammar)
    - Recursively Enumerable (Type-0 Grammar)
      - Modern Syntactic Grammars (minimalism, e.g.)
- Grammar *generates*, Machine *recognizes (accepts/rejects)*



REGULAR LANGUAGES (LAST PASS, NO REALLY)

- Operations on multiple regular languages ($L_1, L_2 \ldots L_n$)
  - If L_1 and L_2 are Regular Languages, then (because of ϵ) the following are Regular Languages as well:
    - $L_1 \cdot L_2$ (concatenation)
    - $L_1 \mid L_2$ (disjunction/union)
    - L_1^* (Kleene closure, *,+)
  - Additional operations available (intersection, difference, complementation, reversal, see pg. 39 for details)