Formal Language Theory 1

A fundamental question of modern linguistics:

(1) How do speakers of a natural language produce and understand novel expressions?

(2) President Trump was concerned that his pet platypus wasn't drinking enough milk.

Modern linguists' response:

(3) Speakers have internalized a **grammar** for their language.

A grammar G consists of

1. a set of lexical items - meaningful words and morphemes- and,
2. some rules that allow us to combine lexical items to forma arbitrarily many complex expressions.

(4) $G = \langle Lex_G, R_G \rangle$

Grammars for natural languages must be **sound** and **complete**.

(5) a. **Sound:** G generates **only** expressions judged grammatical.
b. **Complete:** G generates **all** expressions judged grammatical.

The **language** generated by a grammar G, L_G, is the closure of Lex_G under R_G.

- Applying the rules in R to the elements of Lex, and then applying the rules to the result, and the applying the rules to the result etc. etc.

$\therefore$ A language is a set of expressions.

- Since Ls are sets, we can apply usual set-theoretic operations on them ($\cup$, $\cap$, $-$).

Grammars give us ways of defining languages.

1 String languages

We start with a set Σ, the **alphabet**.

(6) Simple alphabet:
$\Sigma = \{a, b, c\}$

- The **words** on Σ are the finite strings on the elements of Σ. (a, $abaa$, acb, etc.)
- The empty string ϵ is a word.
- Σ^* is the set of all the words on Σ. (Kleene star)
- A (string) language L is a subset of $L \subseteq \Sigma^*$.

(7) Examples of languages
a. $L_1 = \{\epsilon, a, b, c, aa, ab, ba, bb, bc, ca, cb, cc\}$. This is the set of words of length at most 2.

b. $L_2 = \{a, aa, aaa, aaaa, \ldots a^n, \ldots\}$. words of length 1 consisting of all as.
c. $a^n b^n = \{$ab, aabb, aaabbb, aaaabbbb, $\ldots\}$

(8a) is a **finite** language, so if we want to define it, we can just list it.

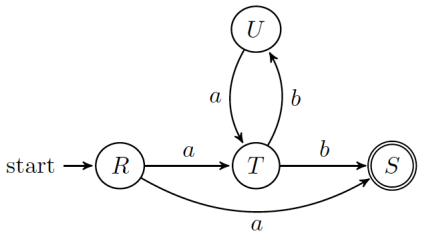
- (8b-c) are infinite languages.

Three (equivalent) ways of defining (some) infinite languages.

1. Finite state autonoma. (machines)
2. Regular expressions. (pattern)
3. Regular grammars. (grammar)

2 Finite State Automata

$\mathcal{A}$



R = start state. S = end/accepting state.

$\mathcal{A}$ accepts the strings that are constructed by starting at the start state, following the arrows, and getting to the end state.

- L($\mathcal{A}$) = the strings accepted by $\mathcal{A}$.

(8) Accept?
a. a
b. b
c. aa
d. ab
e. ababab
f. abaa

Définition 2.1 *An FSA over* Σ $A = \langle Q, \Sigma, \delta, q, F\rangle$*, where*

1. Q *= a set of states* $\{q_0, q_1, q_2 \ldots q_n\}$
2. Σ *= a finite alphabet.*
3. δ *is a transition function from* $Q \times \Sigma$ *to* $\mathcal{P}(Q)$*.*
4. q_0 *is the start state.*
5. F *= is the set of accepting states* $\{q_m, q_n \ldots\}$*.*

$\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F\rangle$, where

1. Q = {R, T, U, S}

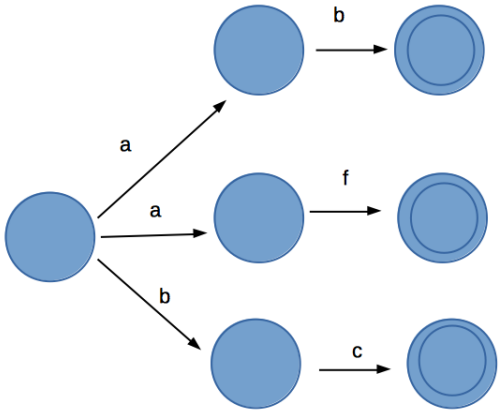
2. $\Sigma = \{a, b\}$
3. F = {S}

(9) a. $\delta(R, a) = \{T, S\}$
b. $\delta(T, b) = \{U, S\}$
c. $\delta(U, a) = \{T\}$

Définition 2.2 *The set of* ***regular languages****, Reg, =* $\{L$*: L(A), for some FSA* $A\}$.

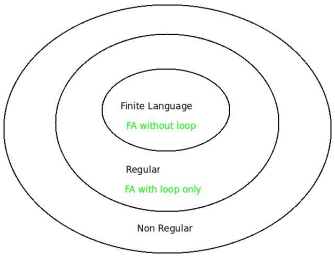
Théorème 2.1 *All finite Ls are regular.* (L_{fin})

Proof sketch: Consider $L = \{ab, af, bc\}$



Some regular languages are infinite (eg. a^n), so we have a hierarchy...

- The set of formal languages has structure!



Are there other subclasses of languages?

(10) $\mathcal{P}(\Sigma^*)$ denotes all the languages over Σ.

(11) Queries
a. $\mathcal{P}(\Sigma^*) = Reg$?
b. Natural languages: NL $\subseteq Reg$?

Proposal: $NL \not\subseteq Reg$.

- To argue in favour of our proposal, we will show that English (ENG) $\notin Reg$.