Arab Republic of Egypt
Ministry of Communications and Information Technology

Information Technology Institute

EPITA
SCHOOL OF ENGINEERING & COMPUTER SCIENCE

A.I. IN AUDIO & SIGNAL PROCESSING

Session 4: Automata and transducers

COURSE STRUCTURE

Quick Summary

Audio processing for AI

- Signal, audio, speech encoding (3h)
- Deep learning for audio processing (4h)

Automata for language modelling

- HMM for speech processing (3h)
- Automata and transducer (3h)

Towards speaking with an AI-bot

- Speech synthesis (4h)
- Automatic speech recognition (4h)
- Speaker and emotion recognition (4h)

SESSION 4: AUTOMATA AND TRANSDUCERS

Quick Summary

1. **Boolean automata**

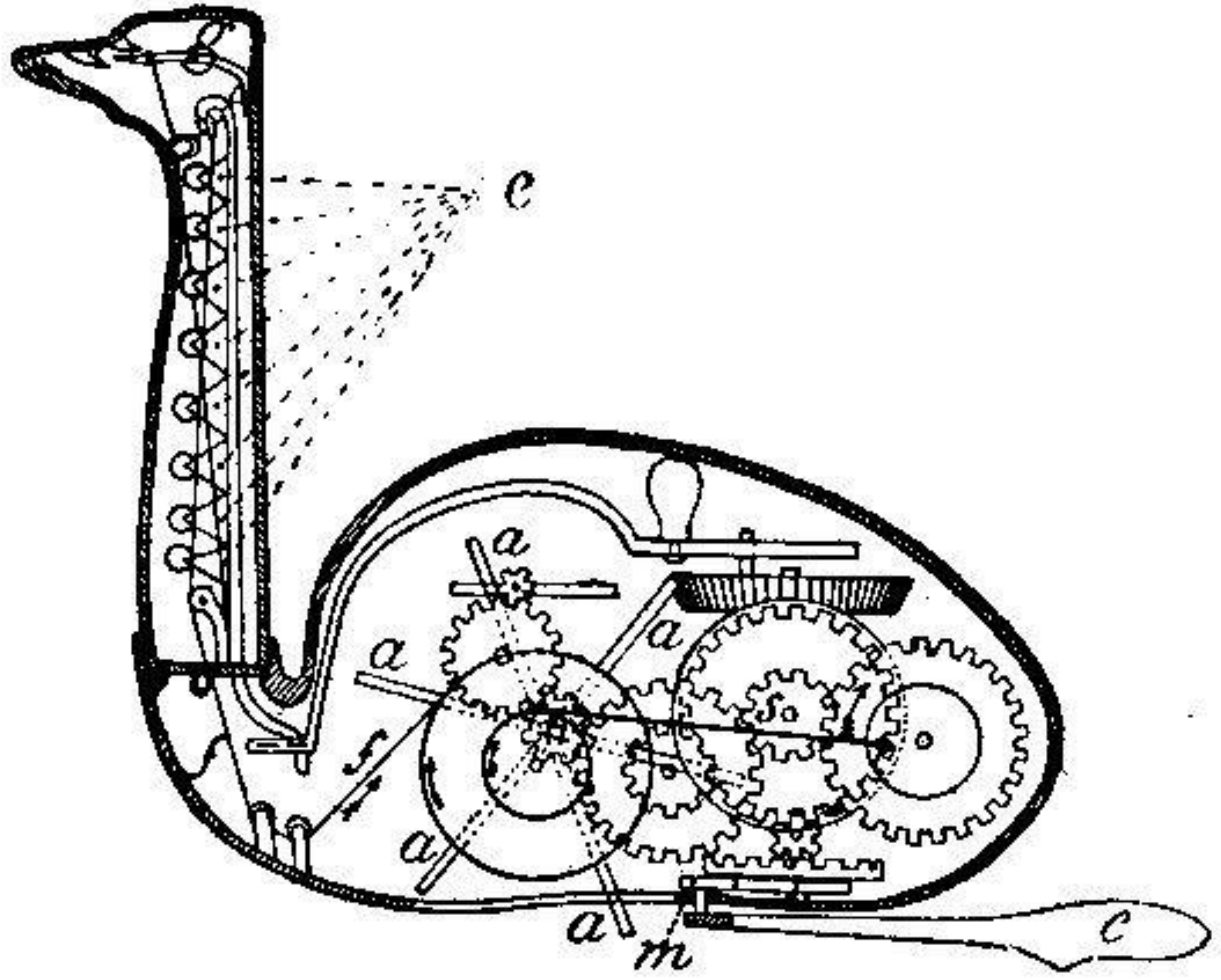
    a) Automata
    b) Transducers
    c) Probabilistic automata

2. **Application to Natural Language Processing**

    a) Lexicon
    b) Morphology and phonology
    c) Part-of-speech tagging
    d) Syntax
    e) Translation

AUTOMATA AND TRANSDUCERS.

Boolean automata and transducers



Le canard : *a*, roues à palette ; *c*, patte palmée ; *e*, châssis, *f*, corde à boyau ; *g*, ressort ; *m*, levier horizontal ; *s*, pignon.

Vaucanson automata (1739)

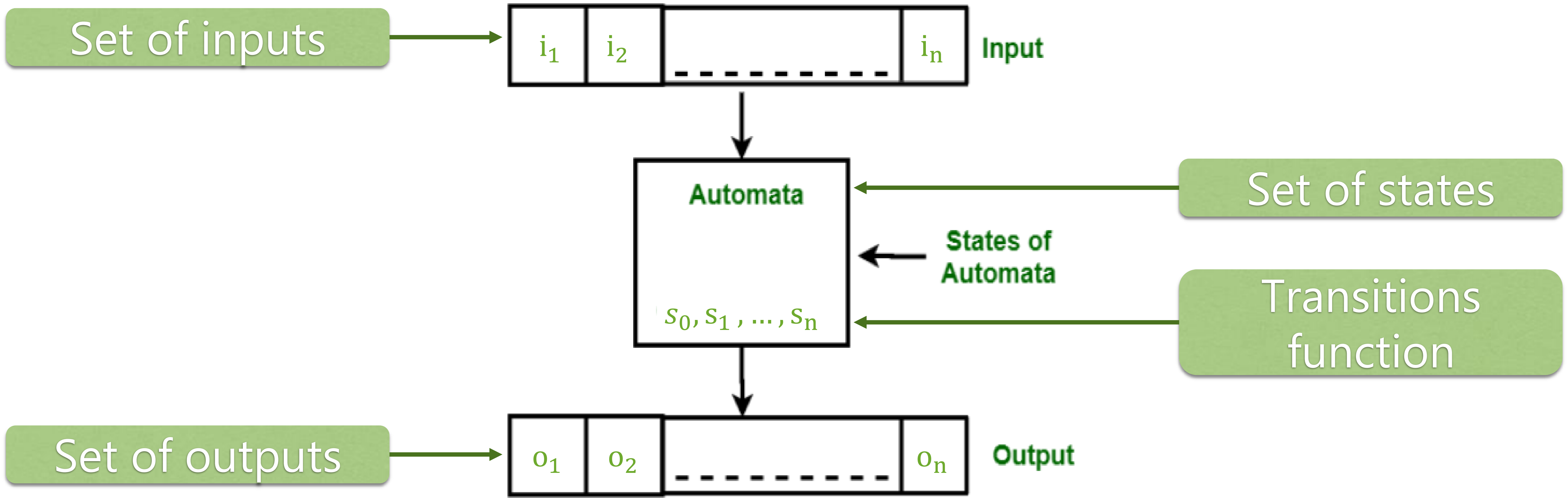
BOOLEAN AUTOMATA AND TRANSDUCERS

Deterministic Finite-state Automata

A deterministic finite-state automaton (DFA) is a 5-tuple $(\Sigma, S, \delta, s_0, F)$, where:

- Σ : *alphabet*, a finite set of symbols
- S : set of states
- δ : transition function $\delta : S \times \Sigma \rightarrow S$
- s_0 : *initial state*
- F : *final states*, a subset of S

$$\text{with} \begin{cases} i_1, i_2, \ldots, i_n \in \Sigma^n \\ s_0, s_1, \ldots, s_n \in S^n \\ s_0 \ initial\ state \\ o_1, o_2, \ldots, o_n \ booleans \end{cases}$$



<u>Example of simple DFA</u>

- Traffic Light

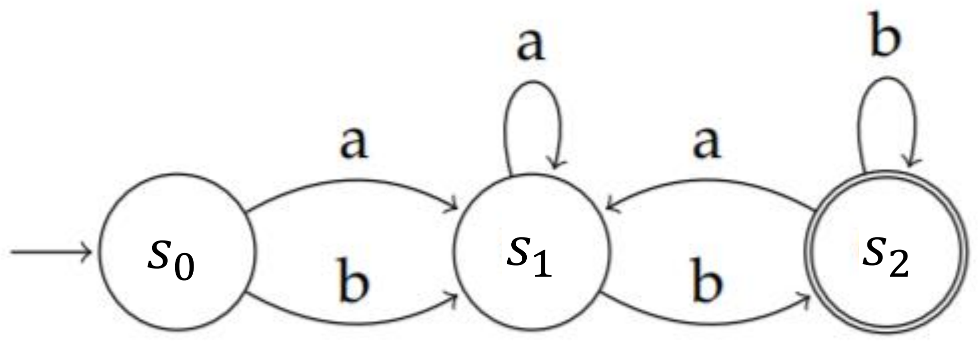
  States: Red, Yellow, Green

  Transitions: After a given time, Red will change to Green, Green to Yellow, and Yellow to Red

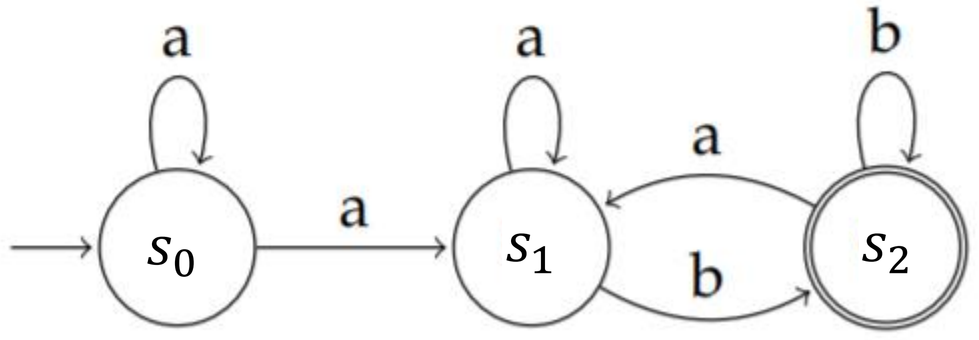
Example of mechanical DFA

BOOLEAN AUTOMATA AND TRANSDUCERS

Non-deterministic Finite-state Automata



deterministic finite automaton.



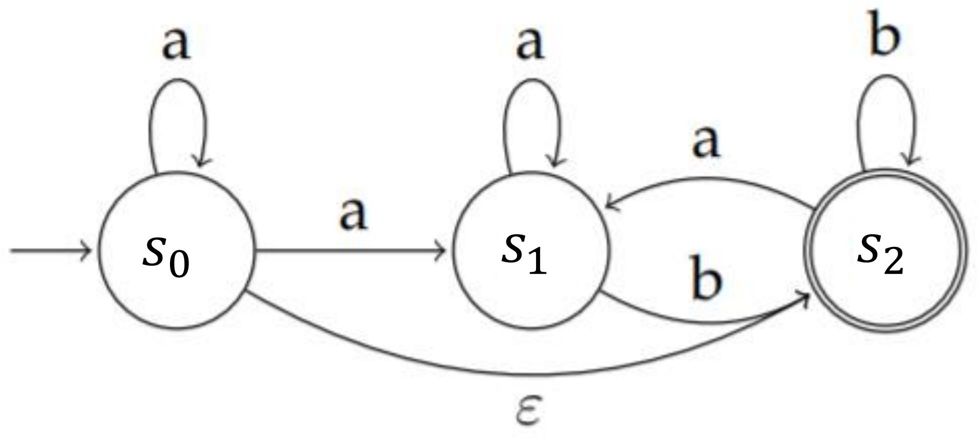
nondeterministic finite automata.

A non-deterministic finite-state automaton (NFA) is a 5-tuple $(\Sigma, S, \delta, s_0, F)$, with the difference in the definition of δ:

- δ : transition function $\delta : S \times \Sigma \to P(S)$

A non-deterministic finite-state automaton with **ϵ-transitions** (ϵ-NFA) is a quintuple $(\Sigma, S, s_0, \delta, F)$, with the difference in the definition of δ:

- δ : transition function $\delta : S \times (\Sigma \cup \{\epsilon\}) \to P(S)$



nondeterministic finite automata with ε-transitions.

with $\begin{cases} a, b \in \Sigma^2 \\ s_0, s_1, s_2 \in S^3 \\ s_0 \ initial\ state \\ s_2 \in F \\ \epsilon \ empty\ symbol \end{cases}$

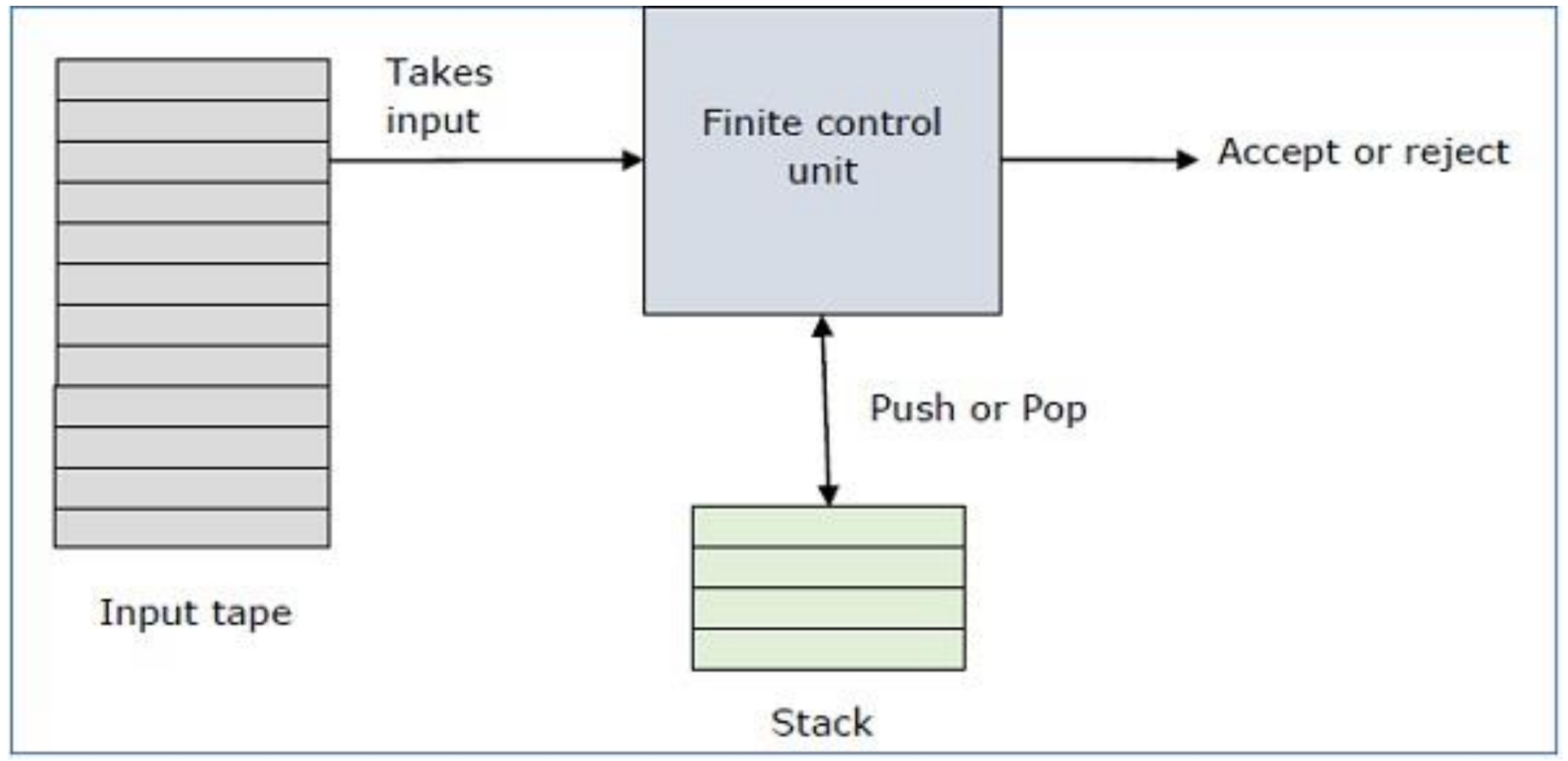
BOOLEAN AUTOMATA AND TRANSDUCERS

Push-down Automata

A push-down automaton (PDA) is a variation of FSA using a stack. It is a 6-tuple $(\Sigma, S, \delta, s_0, \Gamma, F)$, where:

- Σ : *alphabet*, a finite set of symbols
- S : set of states
- δ : transition function $\delta : \; S \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow S \times \Gamma^*$
- s_0 : *initial state*
- Γ : stack which can hold symbols of Σ; Γ^*, string of symbols in stack
- F : *accepted states*, a subset of S

(an initial stack symbol can be added, otherwise we suppose it is empty;
the stack Γ can have its own set of symbols, different from Σ)



basic idea of PDA principle

BOOLEAN AUTOMATA AND TRANSDUCERS

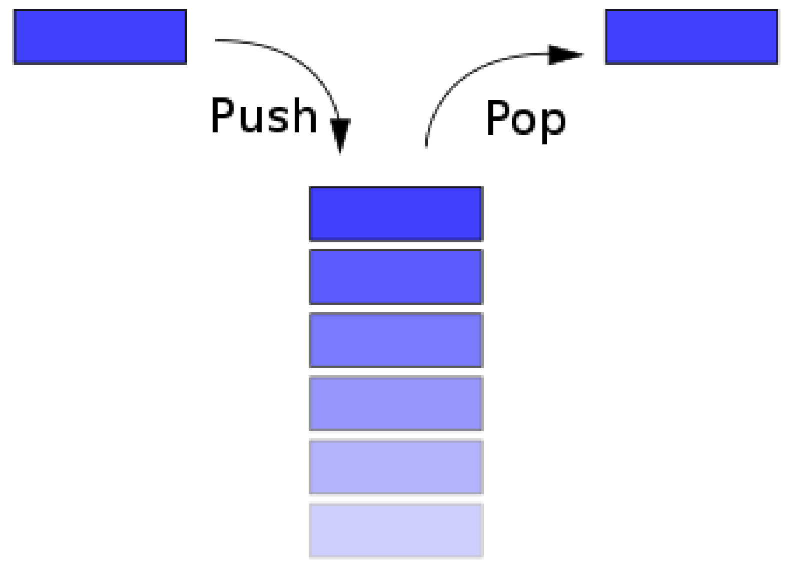
Push-down Automata

Transition process:

- $\delta(s, i, x) = (s', x')$, s' new state, x'string of symbols
- If $x' = \epsilon$, the stack is popped
- If $x' =$ x, the stack is unchanged
- If $x' =$ yzy, then x is removed from stack and replaced by yzy

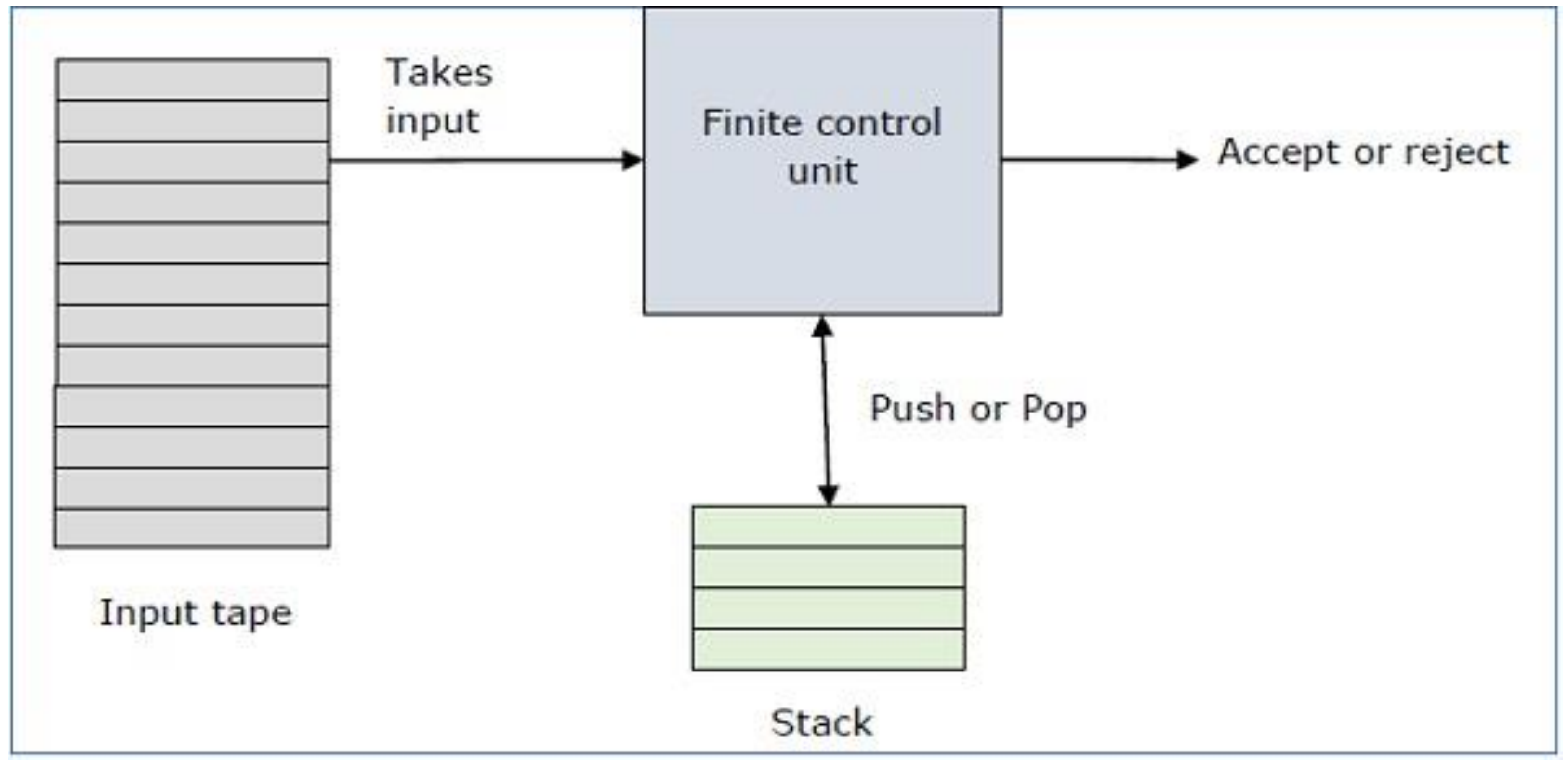
(top of stack)

(read and remove)

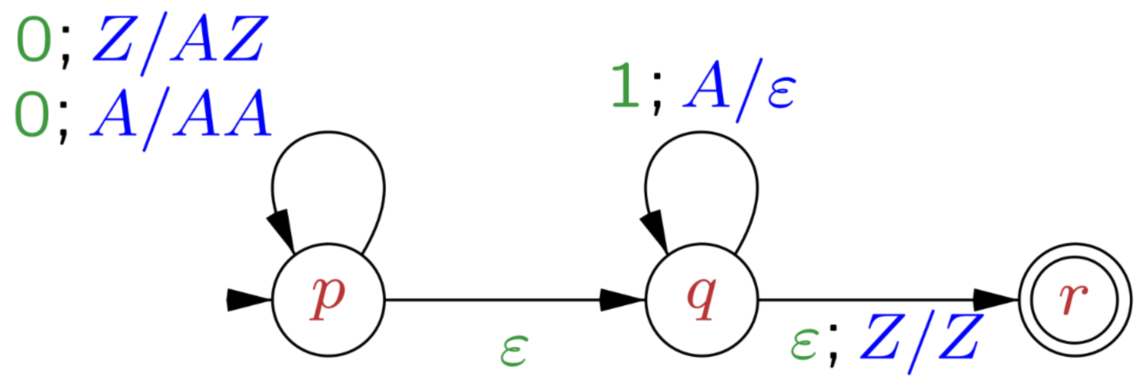


stack Γ

$$\text{with} \begin{cases} s, s' \in S^2 \\ i \in \Sigma \ or \ i = \epsilon \ \ (empty\ symbol) \\ x, x' \in \Gamma^2, \text{strings of elements } of\ \Sigma \end{cases}$$



basic idea of PDA principle



BOOLEAN AUTOMATA AND TRANSDUCERS

Automata optimization

- Determinazation
  → transfrom a non-derterministic automata into a deterministic one

- Minimization
  → reduce the size of an automata

J. Sakarovitch. Eléments of automata theory. 2003.
J. E. Hopcroft, J. D. Ullman. Introduction to Automata Theory, Languages, and Computation. 1979.

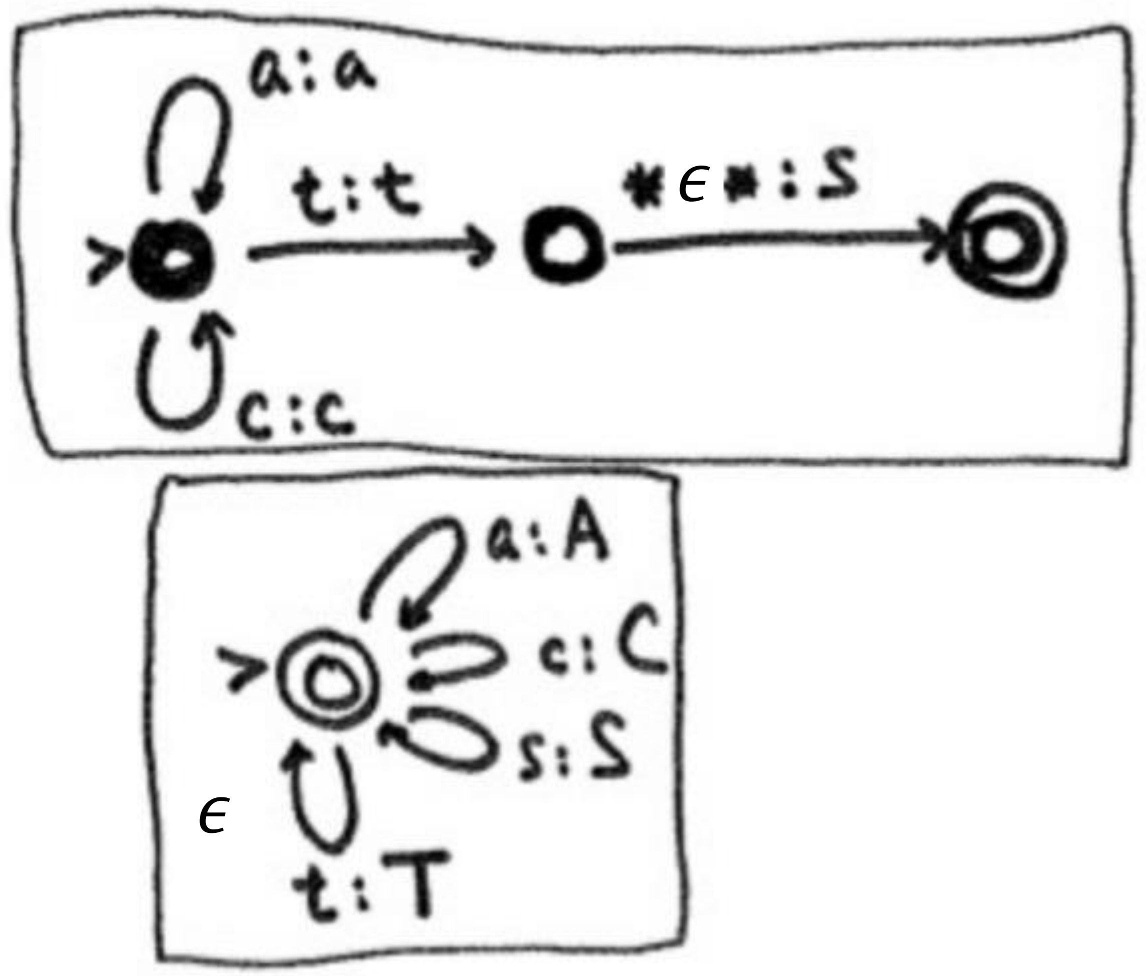
BOOLEAN AUTOMATA AND TRANSDUCERS

Finite-state transducers

A sequential string-to-string transducer (FST) is 6-tuple $(\Sigma, \Delta, S, \delta, s_0, F)$, where:

- Σ : *input alphabet*, a set of symbols
- Δ : *output alphabet*, a set of symbols
- S : set of states
- δ : transition function $\delta : S \times (\Sigma \cup \{\epsilon\}) \to S \times (\Delta \cup \{\epsilon\})$
- s_0 : *initial state* $\in S$
- F : *accepted states*, a subset of S

ϵ is the empty character



Examples:
- Text-to-sound
- Translator
- Part-of-speech tagging

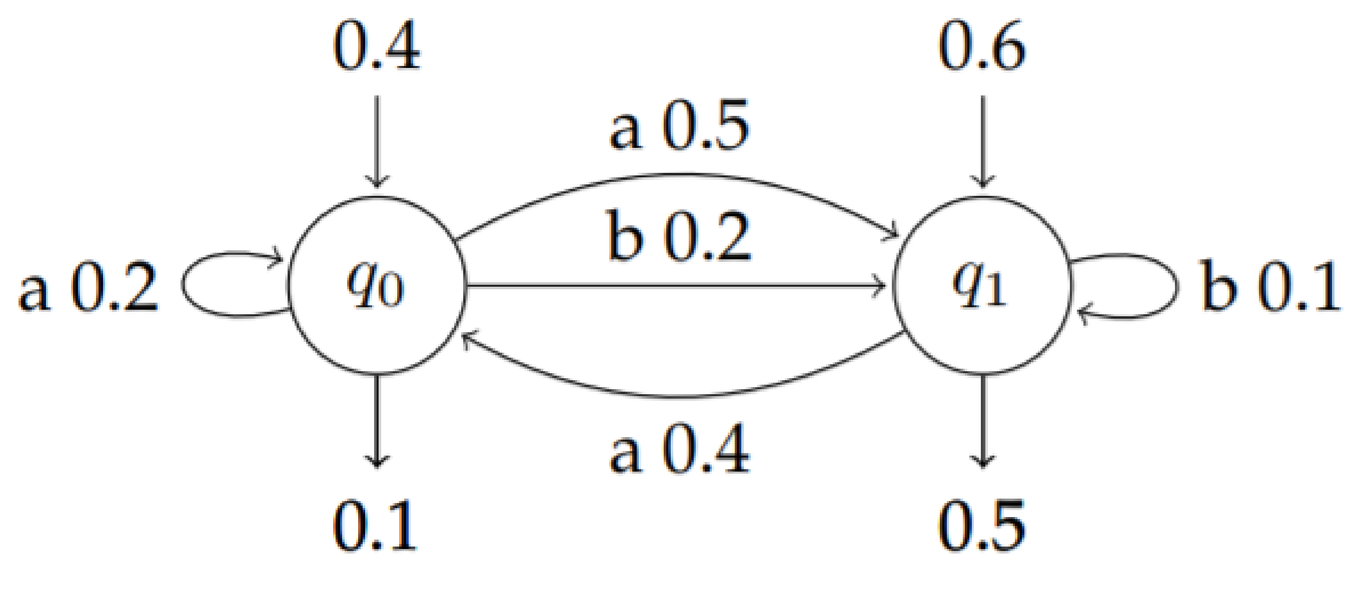
M. Mohri. On Some Applications of Finite-State Automata Theory to Natural Language Processing. 1996.
M. Mohri. Minimization of sequential transducers. 1996.

BOOLEAN AUTOMATA AND TRANSDUCERS

Probabilistic automata

A probabilistic automaton (PA) is defined by a 5-tuple $(\Sigma, S, \phi, \iota, \tau)$, where:

- Σ : *alphabet*, a set of input symbols
- S : set of states
- ϕ : a mapping defining the transition probability function, $\phi : S \times \Sigma \times S \to [0,1]$
- ι : a mapping defining the initial probability of each state, $\iota : S \to [0,1]$
- τ : a mapping defining the final probability of each state, $\tau : S \to [0,1]$



probabilistic automaton

Constraints to be verified:

- $\sum_{s \in S} \iota(s) = 1$
- $\forall s \in S,\ \tau(s) + \sum_{x \in \Sigma} \sum_{s' \in S} \phi(s, x, s') = 1$

Differences between HMM and probabilistic automata:

→ It has been proven that probabilistic automata with no final probabilities are equivalent to hidden Markov models.

Dupont, P., Fran c. D., and Esposito, Y. (2005). Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms. Pattern Recognition, 38(9):1349–1371

AUTOMATA AND TRANSDUCERS.

Application to Natural Language Processing

APPLICATION TO NATURAL LANGUAGE PROCESSING

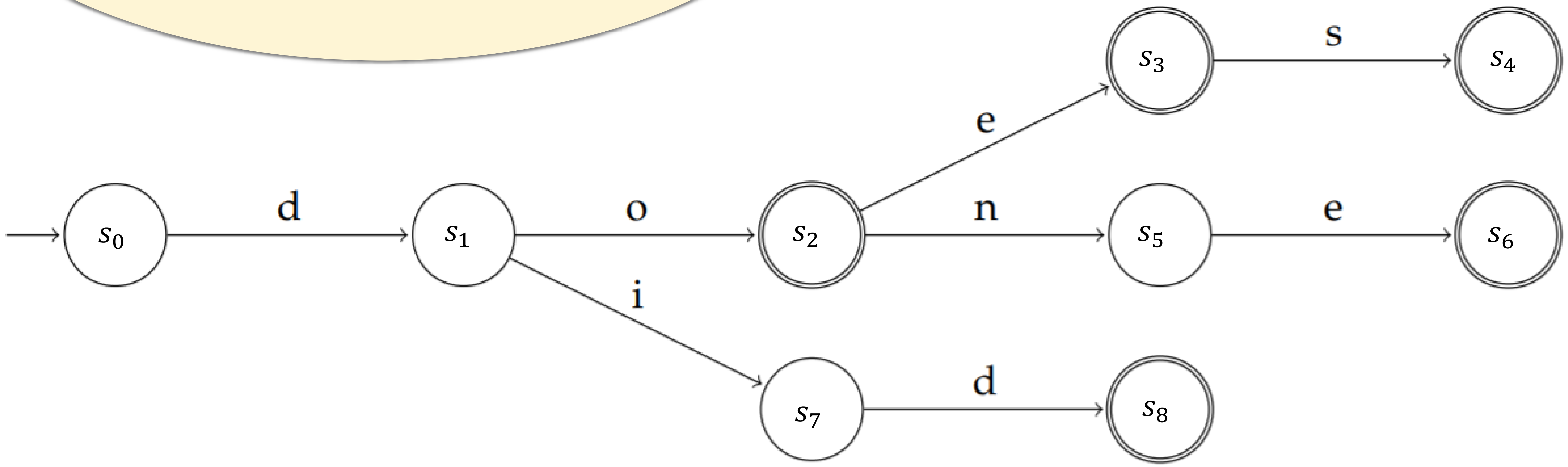
Lexicon

- Context

  did,V:PRET
  do,N:s
  do,V:INF
  doe,N:s:p
  does,V:P3s
  does,N:p
  done,V:PP done,A

- listings to automata

Lexicology
Study of words, their nature, their function as symbols, their meaning, the relationship of their meaning to epistemology



lexicon represented by an automaton.

Lexicon	FDELAF	GDELAF	EDELAF
Initial size	21.2Mb	27.9Mb	3.6Mb
Automata size	1.2Mb	3.1Mb	470Kb

Lexicon representations: memory consumption. (Mohri, 1996b)

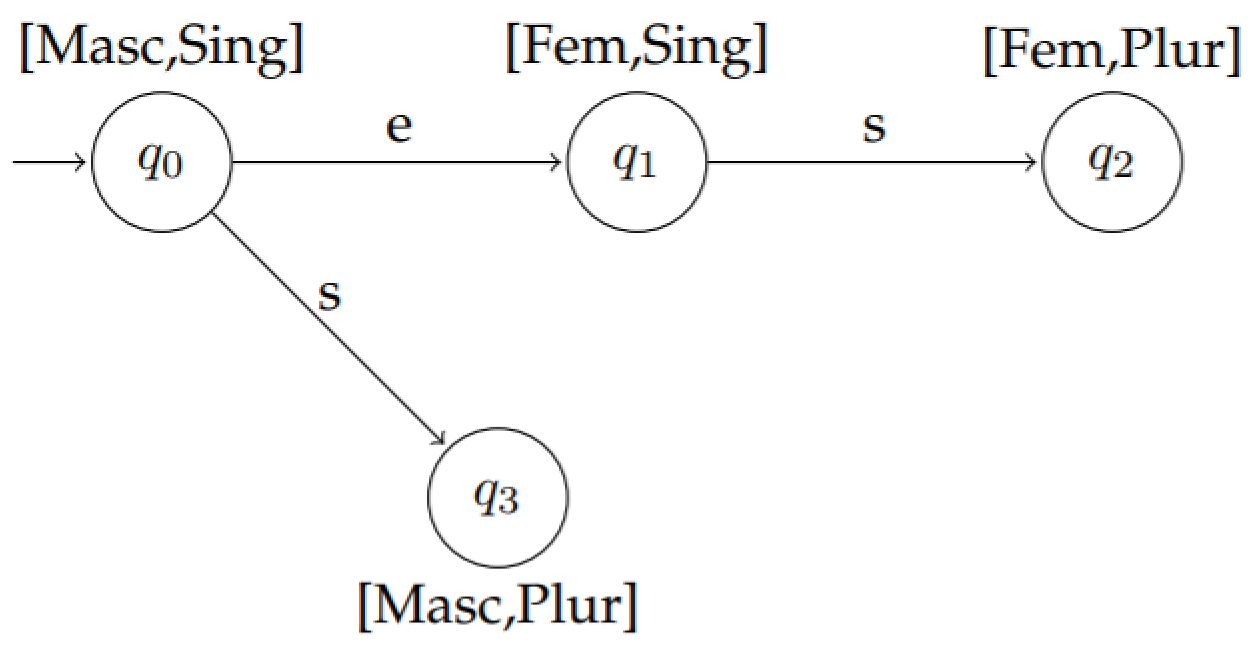
Revuz, D. (1991). Dictionnaires et lexiques. Méthodes et algorithmes. PhD thesis

APPLICATION TO NATURAL LANGUAGE PROCESSING

Morphology and phonology

- N+“s” for the plural (dog, dogs),
- V+“s” for the third form of the present (read, reads),
- V+“ed” for the past participle of most verbs (match, matched),
- V+“er” to designate the one who (to hunt, the hunter),
- Adj+“ly” to qualify a way of doing things (slow, slowly)
- Political figure+“ism” to refer the political party of this figure (Bush, bushism)

Morphological rules need to be implemented in the automaton to increase its efficience.



Sample of morphological inflection rules of adjectives in French.

Phonology
syllable, onset and rime, articulatory gestures, articulatory features, mora,… sound are structured to convey linguistic meaning

Morphology
Field of linguistics that focuses on the study of the internal structure of a word. It is the smallest units of syntax.

AUTOMATA AND TRANSDUCERS.

Thank you for your attention.

References:

- Jimmy Ma