Finite state transducers, String-to-Weight Transducers and their application in speech processing

Miloš Janda, ijanda@fit.vutbr.cz

Department of Computer Graphics and Multimedia
Faculty of Information Technology
Brno University of Technology

December 19, 2010

Outline

Outline

Outline

Outline

Outline

Definition

Speech processing is the study of speech signals and the processing methods of these signals

Categories of speech processing

- Speech recognition (LVCSR)
- Keyword spotting (KWS)
- Languge (LID) and speaker (SID) identification
- Speaker verification
- Phoneme detection

Motivation for Automatic Speech Recognition (ASR)

Spech is main form of human communication

Aplications

- Military (LID, SID, KWS)
- Telephony and other domains
- Remote controlling
- Teaching (Dictionaries, Pronunciation dicts)
- ...

Simple schema of ASR

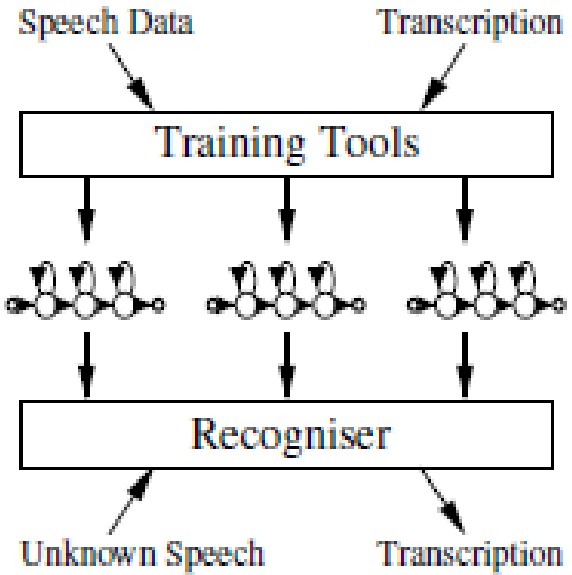

Figure 1: Recognition system schema.

Overall schema of ASR

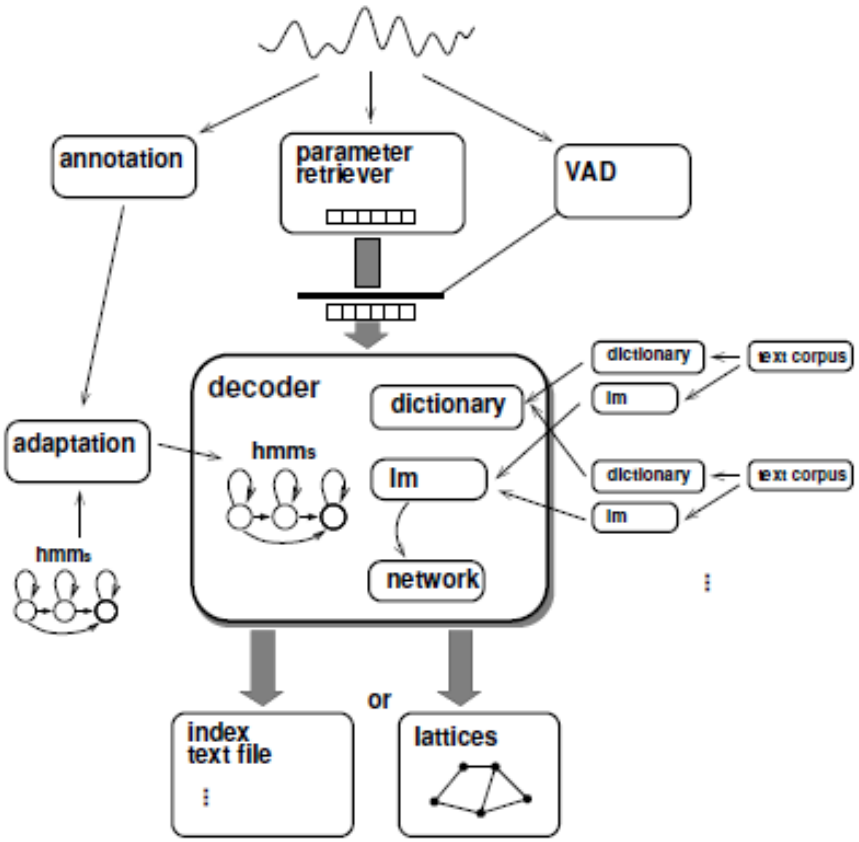

Figure 2: Overall schema of speech recognizer.

Nondeterministic Finite State Automaton (NFSA)

Finite State Automaton (nondeterministic)

Let M is a 5-tuple= $(Q, \Sigma, \delta, i, F)$
where

- Q is a finite set of states
- Σ is an input alphabet
- δ is a transition function that maps state-symbol pairs to set of states:

$$\delta : Q \times \Sigma \rightarrow 2^Q$$

$$(\delta : Q \times \Sigma \cup \{\epsilon\} \rightarrow 2^Q)$$

- i is initial state: $i \in Q$
- F is set of accepting (final) states: $F \subseteq Q$

Deterministic Finite State Automaton (DFSA)

Finite State Automaton (deterministic)

Let M is a 5-tuple= $(Q, \Sigma, \delta, i, F)$
where

- Q is a finite set of states
- Σ is an input alphabet
- δ is a transition function that maps state-symbol pairs to state:

$$\delta : Q \times \Sigma \rightarrow Q$$

- i is initial state: $i \in Q$
- F is set of accepting (final) states: $F \subseteq Q$

Sequential String-to-String Transducer

Sequential String-to-String Transducer

Let T is a 7-tuple= $(Q, \Sigma, \Delta, i, F, \delta, \sigma)$
where

- Q is a finite set of states
- Σ and Δ, finite sets corresponding to input and output alphabets
- $i \in Q$ is the initial state
- $F \subseteq Q$, the set of final states
- δ, the state transition function which maps $Q \times \Sigma \rightarrow Q$
- σ, the output function which maps $Q \times \Sigma \rightarrow \Delta$

Example of Sequential String-to-String Transducer

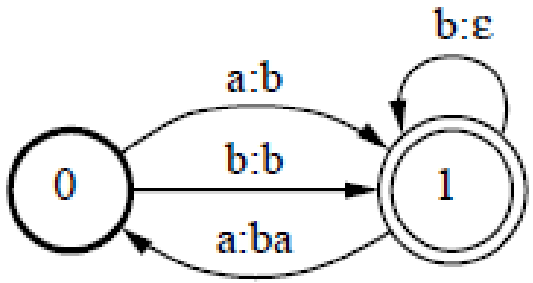

Figure 3: Sequential String-to-String Transducer.

Subsequential and p-Subsequential Transducer

Subsequential Transducer

Generalized sequential transducer
with possibility of generating an additional output string at final states.

p-Subsequential Transducer

Same as Subsequential transducer,
but limited to generate at most p final output strings at each final state.

Example of 2-Subsequential Transducer

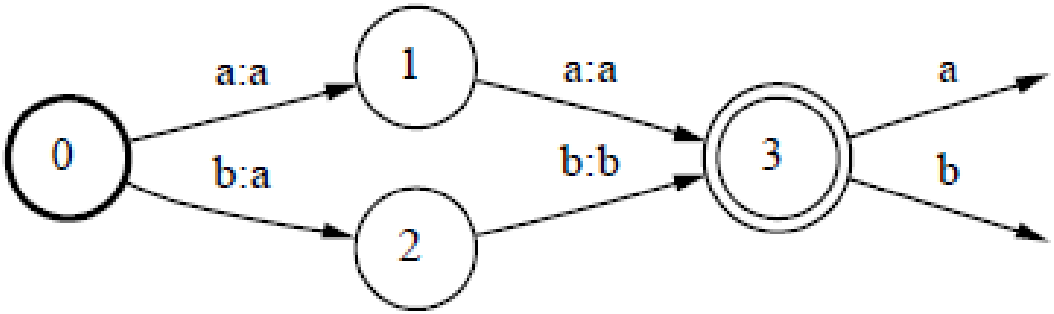

Figure 4: 2-Subsequential Transducer.

String-to-Weight Transducer (Weighted Acceptor - WFSA)

String-to-Weight Transducer

Let T is a 7-tuple= $(Q, \Sigma, i, F, E, \lambda, \rho)$
with

- Q a finite set of states
- Σ the input alphabet
- $i \in Q$ is the initial state
- $F \subseteq Q$, the set of final states
- $E \subseteq Q \times \Sigma \times \mathbb{R}_+ \times Q$ a finite set of transitions
- λ the initial weight function mapping i to $\mathbb{R}_+$
- ρ the final weight function mapping F to $\mathbb{R}_+$

String-to-Weight Transducer (Weighted Acceptor - WFSA)

Transition function

Transition function δ mapping $Q \times \Sigma$ to 2^Q:

$$\forall (q, a) \in Q \times \Sigma,$$
$$\delta(q, a) = \{g' | \exists x \in \mathbb{R}_+ : (q, a, x, q') \in E\}$$

Output function

Output function σ assigning to each transition its weight:

$$\forall t = (p, a, x, q) \in E, \sigma(t) = x$$

String-to-Weight Transducer (Weighted Acceptor - WFSA)

Path

A path π is a sequence of successive transitions from one state to another:

$$\pi = ((q_0, a_0, x_0, q_1), ..., (q_{m-1}, a_{m-1}, x_{m-1}, q_m)$$
$$\text{with } (q_i, a_i, x_i, q_{i+1}) \in E$$

The label of a path π is $a_0 ... a_{m-1}$.

The set of paths form q to q' labeled with input string w is abbreviated by:

$$q \rightsquigarrow^{w} q'$$

Example of Weighted Acceptor (WFSA)

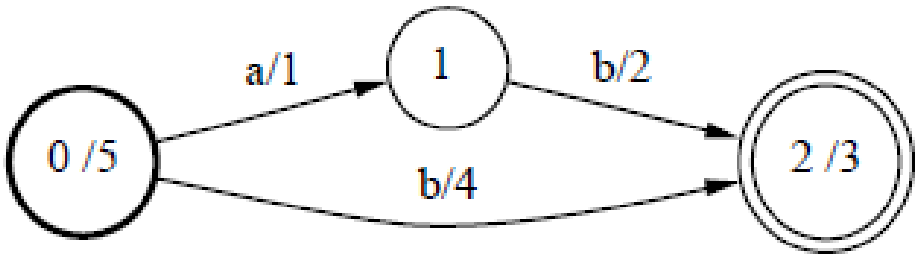

Figure 5: Weighted Acceptors (WFSA).

String-to-Weight Subsequential Transducer (Weighted Transducer - WFST)

String-to-Weight Subsequential Transducer

Let T is a 8-tuple= $(Q, \Sigma, \Omega, i, F, E, \lambda, \rho)$
with

- Q a finite set of states
- Σ the input alphabet
- Ω the output alphabet
- $i \in Q$ is the initial state
- $F \subseteq Q$, the set of final states
- $E \subseteq Q \times \Sigma \times \Omega \times \mathbb{R}_+ \times Q$ a finite set of transitions
- λ the initial weight function mapping i to $\mathbb{R}_+$
- ρ the final weight function mapping F to $\mathbb{R}_+$

Example of Weighted Transducer (WFST)

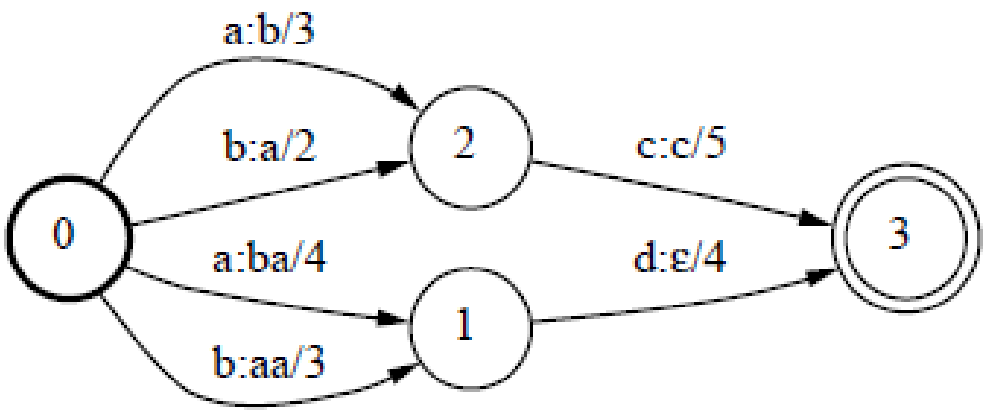

Figure 6: Weighted Transducer (WFST).

Determinization of String-to-Weight Transducers

Determinization

The determinization algorithm takes a non-subsequential string-to-weight transducer
$T_1 = (Q_1, \Sigma, I_1, F_1, E_1, \lambda_1, \rho_1)$
and transform it into an equivalent, subsequential string-to-weight transducer
$T_2 = (Q_2, \Sigma, i_2, F_2, \delta_2, \sigma_2, \lambda_2, \rho_2)$.

Multiple states reachable from one state on the same input symbol are merged.
The algorithm begins with the initial states of T_1 and build new states for T_2 by iterating over non-marked states.

Algorithm for determinization

1 $\lambda_2 := \min_{i \in I_1} \lambda_1(i)$
2 $i_2 := \bigcup_{i \in I_1} \{(i, \lambda_1(i) - \lambda_2)\}$
3 initially , i_2 is the only state in Q_2 and it is unmarked
4 **while** there is an unmarked state q_2 in Q_2 **do begin**
5 mark q_2
6 **if** there exists (q, x) in q_2 such that $q \in F_1$ **then**
7 add q_2 to F_2
8 $\rho_2(q_2) := \min_{q \in F_1, (q,x) \in q_2} (x + \rho_1(q))$
9 **for each** input symbol a and transition t such that $out(q_2, a, t) \neq \emptyset$ **do begin**
10 $\sigma_2(q_2, a) := \min_{(q,x,t) \in out(q_2,a)} x + \sigma_1(t)$
11 $\delta_2(q_2, a) := \bigcup_{q' \in reachable(q_2,a)} \{(q', \min_{(q,x,t) \in out(q_2,a), dest(t)=q'} (x + \sigma_1(t) - \sigma_2(q_2, a))\}$
12 **if** $\delta_2(q_2, a)$ is not in Q_2 **then**
13 add $\delta_2(q_2, a)$ as an unmarked state to Q_2
14 **end**
15 **end**

Figure 7: Derminization algorithm for String-to-Weight Transducer.

Example of determinization

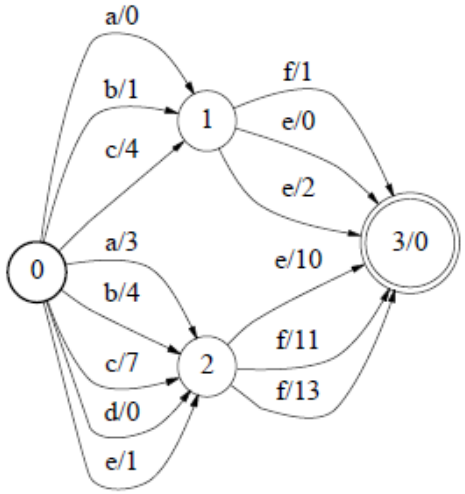

Figure 8: Non-deterministic weighted acceptor.

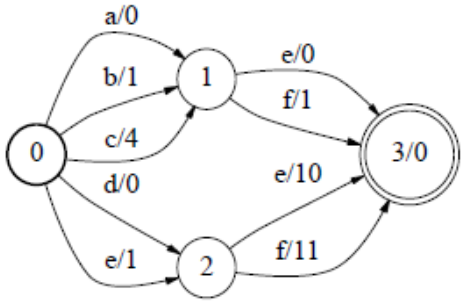

Figure 9: Equivalent WFSA obtained by determinization.

Minimization of String-to-Weight Transducers

Minimization

Process of minimization of String-to-Weight Transducer consist of two steps:

1. Pushing, that adapts the transducer such that its output weights are spreading differently.
2. Usual minimization (same as for FSMs)

Label pushing

Pushing

We define a function d giving the minimal output necessary to reach a final state.
For a given sebsequential transducer T, for any state $q \in Q$, d is defined by:

$$d(q) = min(\sigma(q, w) + \rho(dest(q, w)))$$

Then pushing is defined by:

- $\lambda' = \lambda + d(i)$
- $\forall (q, a) \in Q \times \Sigma$,
  $\sigma'(q, a) = \sigma(q, a) + d(dest(q, a)) - d(q)$
- $\forall q \in Q, \rho'(q) = 0$

Note that pushing does not change the topology of the transducer.

Example of minimization 1/2

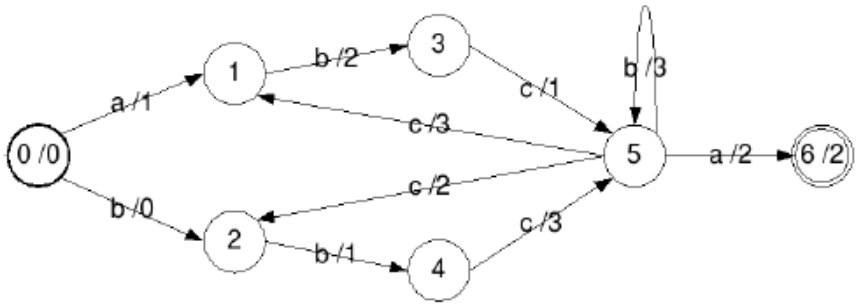

Figure 10: Non-minimal transducer before pushing.

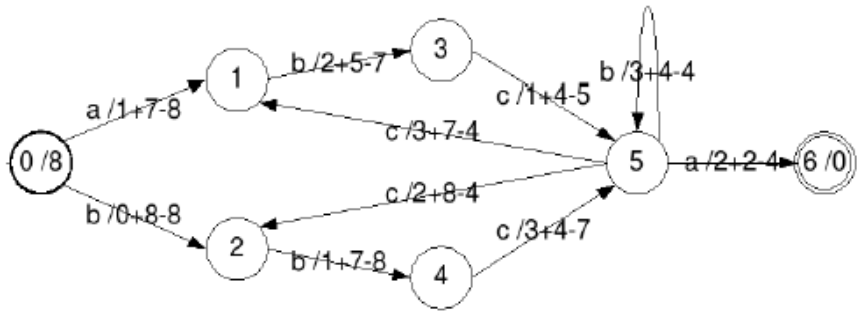

Figure 11: Example of pushing operation.

Example of minimization 2/2

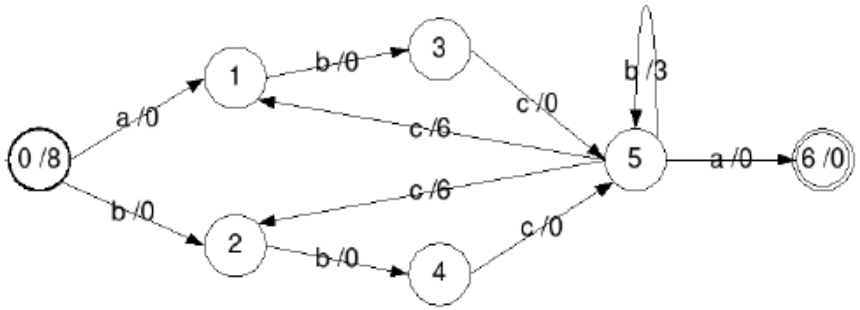

Figure 12: Non-minimal transducer after pushing.

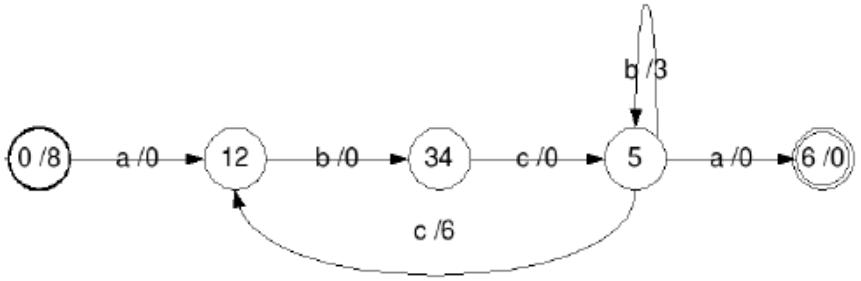

Figure 13: Minimal transducer.

Composition

Composition

The composition of two transducers represent their relational composition.
The composition $T = R \circ S$ of two transducers R and S
has exactly one path mapping sequence u to sequence w for each pair of paths,
the first in R mapping u to v and second in S mapping v to w.

The weight of a path in T is the $\otimes$-product of the weights of corresponding paths in R and S.

Example of composition

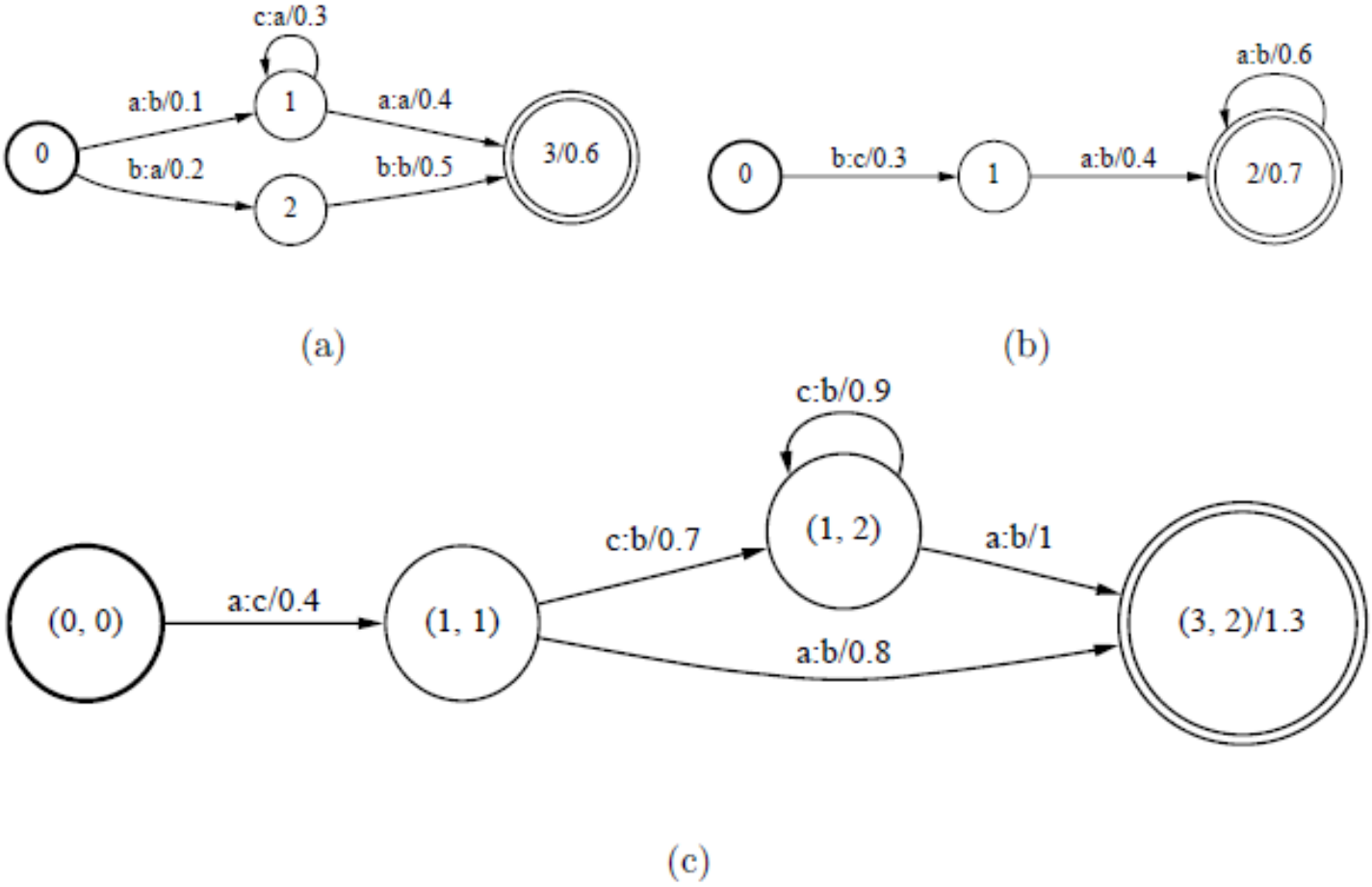

Figure 14: Example of transducer composition.

Speech recognizer composition

Composition

Speech recognizer consist of this parts:

- Acoustic observation sequence of an utterance O
- Acouctic modeling block A
- Context-dependency modeling block C
- Dictionary D
- Grammar G

Thus, the domain of speech recognition can be presented by transducer composition:

$$G \circ D \circ C \circ A \circ O$$

Stages of speech recognition

Figure 15: Stages of speech recognition.

Lattices

Output sentense, Lattice

Sometimes, we are not interested in the best path only,
but we would like to obtain a lattice of hypotheses.
This is achieved by storing more than just the best token in the end-of-word state.
A word-level graph - Lattice - is created this way.

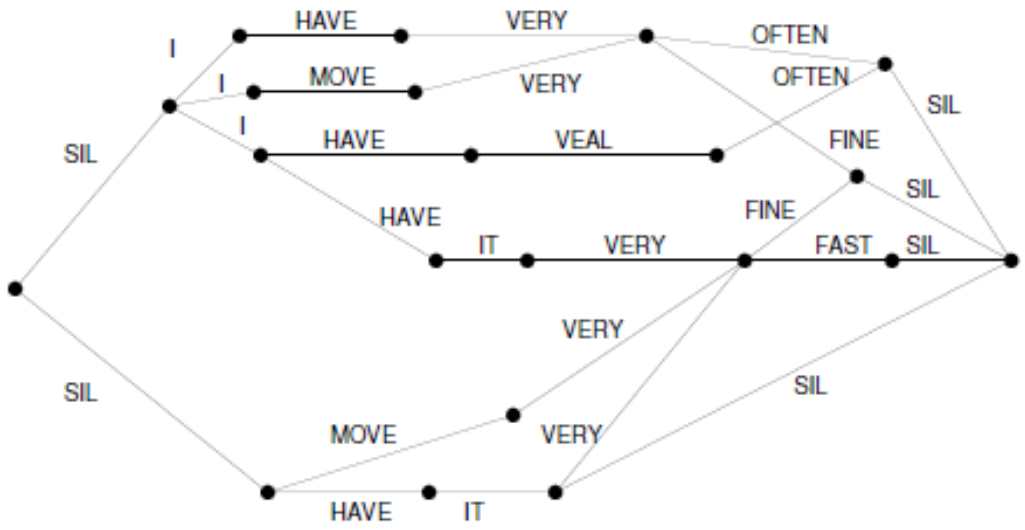

Figure 16: Lattice sample.

Example of lattice

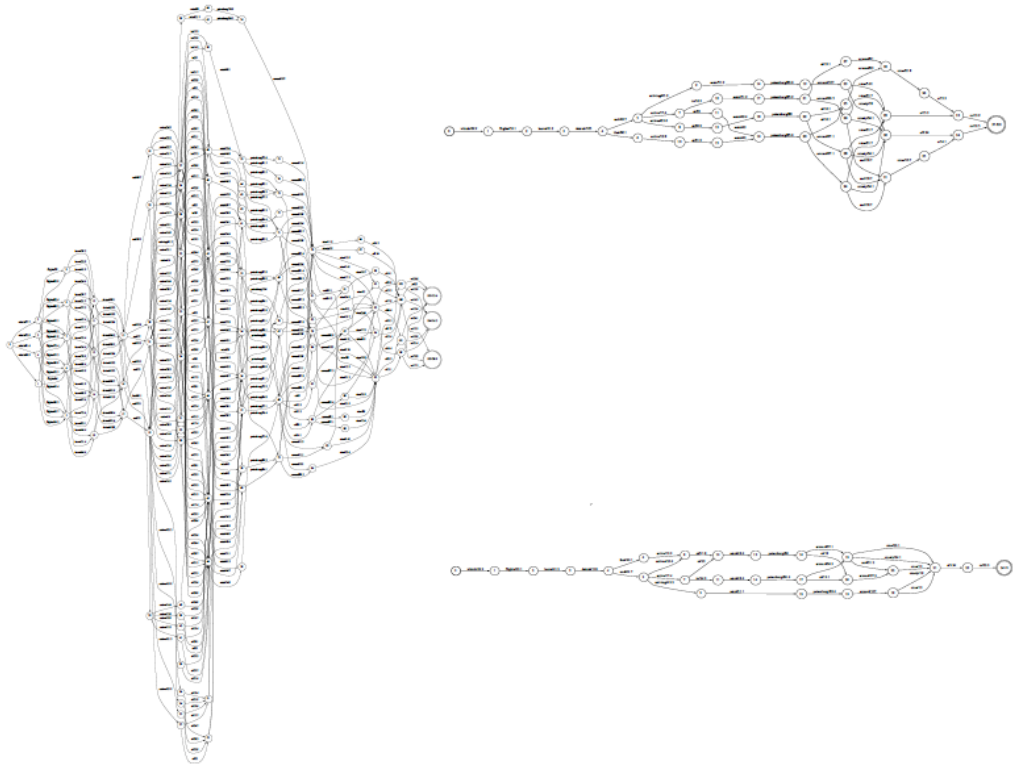

Figure 17: Word lattice W_1, ATIS task, for the utterance Which flights leave Detroit and arrive at Saint-Petersburg around nine am ?

For determinization only we can obtain reduction factor for states ≈ 3 *and for transitions* ≈ 9.

For determ. $+$ *minim. is reduction factor* ≈ 5 *for states and* ≈ 17 *for transitions.*

End of presentation

Thank you for attention

Literature

- Mehryar Mohri, Fernando Pereira, Michael Riley:
  Weighted finite-state transducers in speech recognition.
  Computer Speech Language, Vol. 16, No. 1. (2002), pp. 69-88.

- Mehryar Mohri: Finite-State Transducers in Language and Speech Processing.
  Computational Linguistics, Vol. 23, No. 2. (1997), pp. 269-311.

- Gold Ben, Morgan Nelson: Speech and Audio Signal Processing.
  John Wiley Sons, 2000