Weighted Finite State Transducers

Tom Ko

Finite-state machines

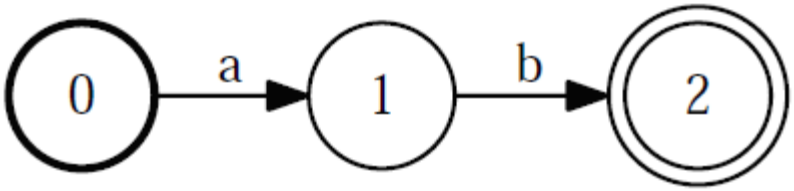
- A **finite-state machine** (**FSM**) or **finite-state automaton** (**FSA**, plural: *automata*) is a mathematical model of computation.

Finite-state acceptors (FSAs)

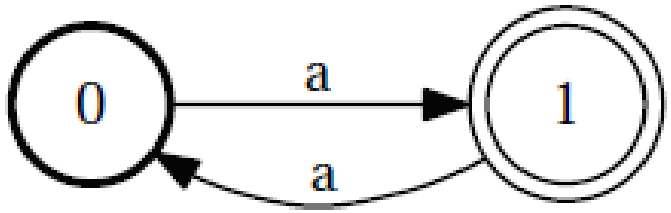
- **Acceptors** produce binary output, indicating whether or not the received input is accepted.
- If it ends the an **final** state after processing a series of inputs, the machine is said to have **accepted** the input; otherwise, it is said to have **rejected** the input.
- Each FSA consists of a set of states, an initial state, a set of final states, and a set of transitions between states.
- Each transition has a source state, a destination state, a label.

FSA – examples

- This FSA only accepts “ab”



- This FSA only accepts string with odd number of ‘a’s



Finite-state acceptors

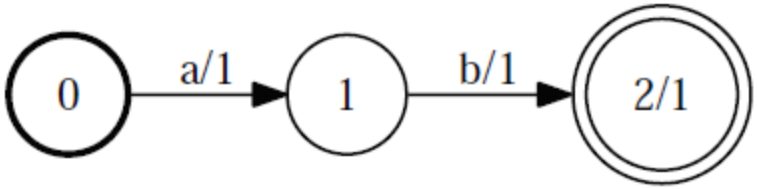
- The string set which an FSA accepts forms a language.
- Thus it is convenient to express languages with FSAs.

Operations on FSAs

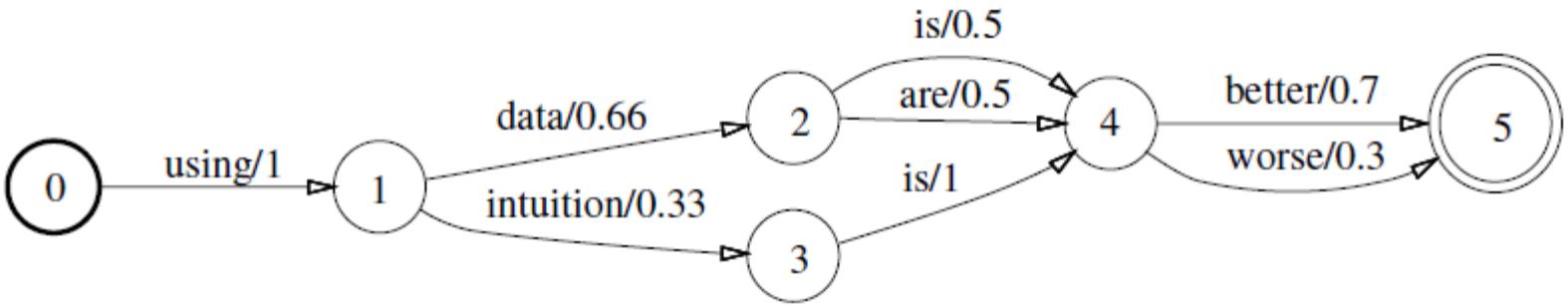
- New FSAs can be obtained by
- **Complement** of an FSA
  - The resulting FSA generates the language which is rejected by the original FSA
- **Union** of two FSAs
  - The resulting FSA generates the language which is accepted by either of the original two FSAs. (or)
- **Intersection** of two FSAs
  - The resulting FSA generates the language which is accepted by both the original two FSAs. (and)
- https://www.youtube.com/watch?v=ZxfaOHs2cX8

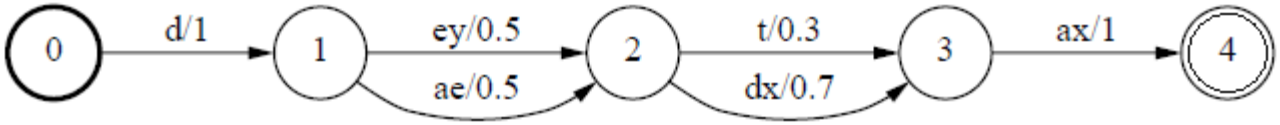
Weighted finite-state acceptors (WFSAs)

- Each WFSA consists of a set of states, an initial state, a set of final states (with final weights), and a set of transitions between states.
- Each transition has a source state, a destination state, a label and a weight.
- Each accepted string is assigned a weight, namely the accumulated weights along accepting paths for that string, including final weights.
  - Weighted Acceptors provides a mapping from symbol sequences to weights.



WFSA – examples

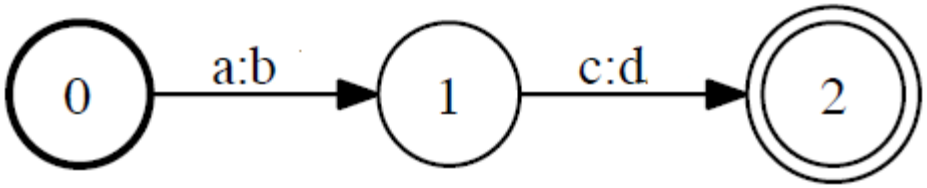




possible pronunciations of “data”

Finite-state transducers

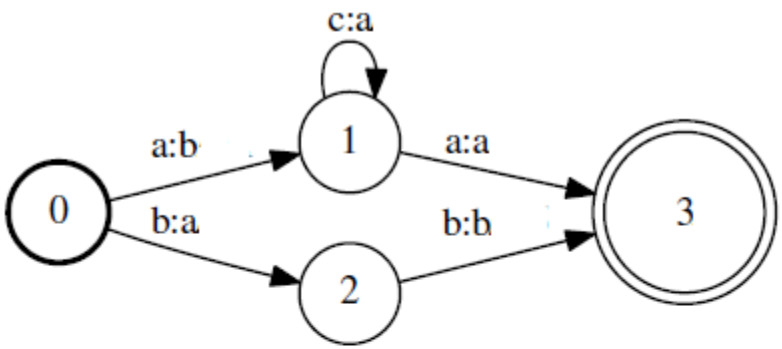
- A finite-state transducer is a finite automaton whose state transitions are labeled with both input and output symbols.
- It can represents mappings from strings to strings
- This FST maps “ac” to “bd”



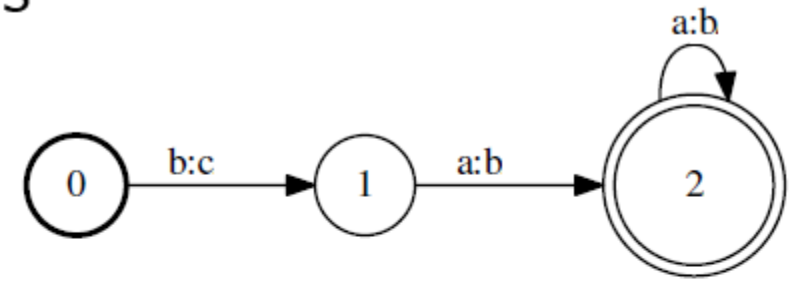
Composition of FST

- A transducer R maps string x to string y.
- A transducer S maps string y to string z.
- The composition of two transducers T = R ∘ S maps string x to z.

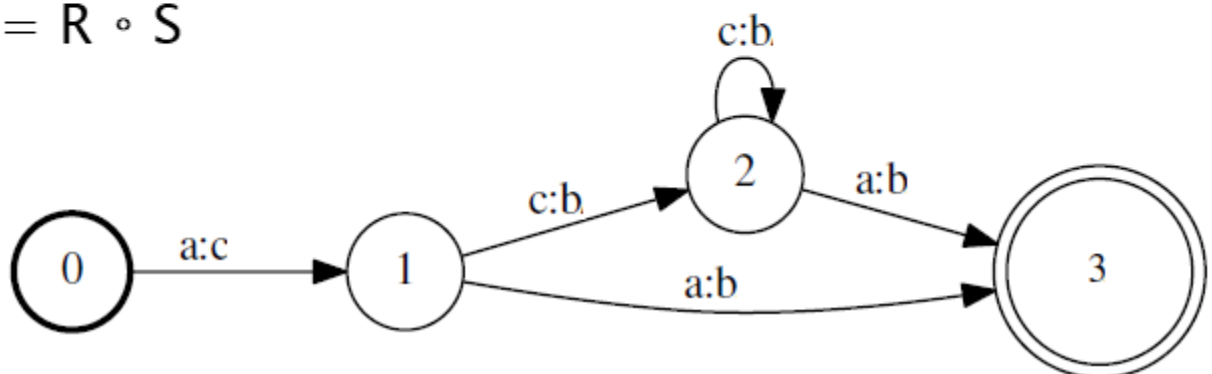
R



S

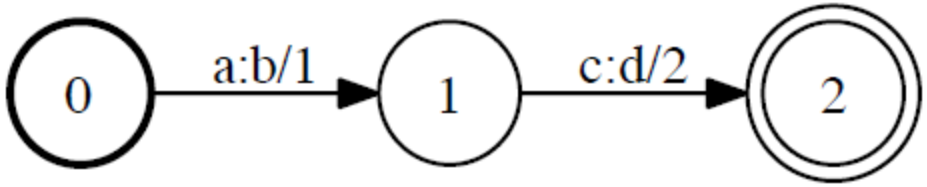


T = R ∘ S



Weighted finite-state transducers

- Weights may encode probabilities, durations, penalties, or any other quantity that accumulates along paths to compute the overall weight of mapping an input string to an output string.



Weighted FST - Semiring

- If the transition weights represent probabilities, the operation to accumulate weights is the product.
- If instead the weights represent log probabilities (common in ASR), the operation is the sum.
- More generally, the weight operations for a weighted transducer can be specified by a *semiring*

Semiring

- In weighted automata, the weights and their binary operations “addition” and “multiplication” are formally defined to generalize the automata and their algorithms.
- In the theory, weights and their operations are defined by a *semiring*, which is an algebraic structure in abstract algebra.
- This means that any kinds of weights can be dealt with in the automata algorithms if a semiring can be defined over the set of weights.

Semiring	Set	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1
Probability	$\mathbb{R}$	$+$	$\times$	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	$+\infty$	0
Tropical	$\mathbb{R} \cup \{-\infty, +\infty\}$	min	$+$	$+\infty$	0

Semiring

- A semiring is defined as $(K, \oplus, \otimes, \bar{0}, \bar{1})$, where K is a set of elements, $\oplus$ and $\otimes$ are two formally defined binary operations, i.e., "addition" and "multiplication," over K, $\bar{0}$ is an additive identity element, and $\bar{1}$ is a multiplicative identity element.

- The semiring must satisfy the axioms listed in Table 3.2
  - (referencing Takaaki Hori, Speech Recognition Algorithms Using Weighted Finite State Transducers)

Table 3.2: Algebraic structure of semiring, that satisfies the listed axioms for all $x, y, z \in \mathbb{K}$

Associativity for addition	$(x \oplus y) \oplus z = x \oplus (y \oplus z)$
Commutativity for addition	$x \oplus y = y \oplus x$
Associativity for mutiplication	$(x \otimes y) \otimes z = x \otimes (y \otimes z)$
Distributivity for multiplication over addition	$(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$ $x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$
Property of additive identity	$\bar{0} \oplus x = x \oplus \bar{0} = x$ $\bar{0} \otimes x = x \otimes \bar{0} = \bar{0}$
Property of multiplicative identity	$\bar{1} \otimes x = x \otimes \bar{1} = x$

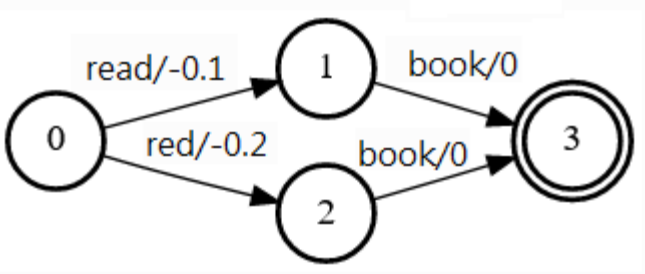
Semiring

- In WFST-based speech recognition, the tropical semiring is mainly used, which consists of a set of real-valued weights with “addition” and “multiplication” defined as the minimum of the two and ordinary addition, respectively.
- In some optimization steps for WFSTs, the log semiring is also used.

Semiring	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Probability	$[0, 1]$	$+$	$\times$	0	1
Log	$[-\infty, +\infty]$	$\oplus_{\log}$	$+$	∞	0
Tropical	$[-\infty, +\infty]$	$\min$	$+$	∞	0
String	$\Sigma^* \cup \{s_\infty\}$	$\wedge$	$\cdot$	s_∞	ε

$\oplus_{log}$ is $x \oplus_{log} y = -log\left(e^{-x} + e^{-y}\right)$

Tropical semiring



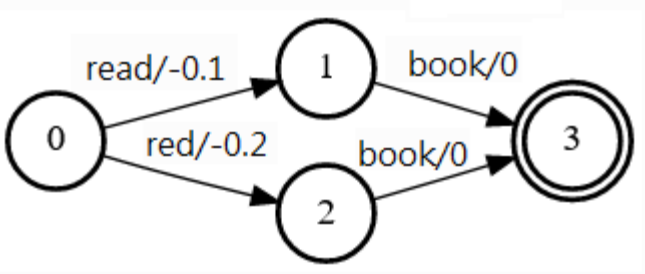
- There are two paths from the start state to the final state:
  - Path 0: state 0 -> state 1 -> state 3, with score: -0.1 + 0 = -0.1
  - Path 1: state 0 -> state 2 -> state 3, with score: -0.2 + 0 = -0.2
- So in the tropical semiring, we would consider that "total score" of these two paths is

$$\min(-0.1, -0.2) = -0.2$$

Semiring	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Log	$[-\infty, +\infty]$	$\oplus_{\log}$	$+$	∞	0
Tropical	$[-\infty, +\infty]$	min	$+$	∞	0

$\oplus_{log}$ is $x \oplus_{log} y = -log\left(e^{-x} + e^{-y}\right)$

Log semiring



- There are two paths from the start state to the final state:
  - Path 0: state 0 -> state 1 -> state 3, with score: -0.1 + 0 = -0.1
  - Path 1: state 0 -> state 2 -> state 3, with score: -0.2 + 0 = -0.2
- So in the tropical semiring, we would consider that "total score" of these two paths is

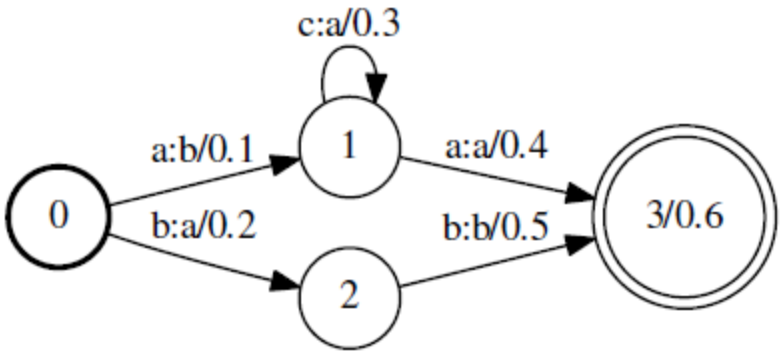
$$-\log(\exp(0.1) + \exp(0.2)) = -0.8444$$

Semiring	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Log	$[-\infty, +\infty]$	$\oplus_{\log}$	$+$	∞	0
Tropical	$[-\infty, +\infty]$	min	$+$	∞	0

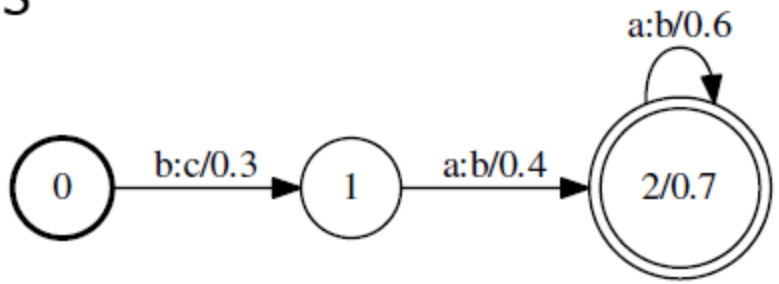
$$\oplus_{log} \text{ is } x \oplus_{log} y = -log\left(e^{-x} + e^{-y}\right)$$

Composition of WFST

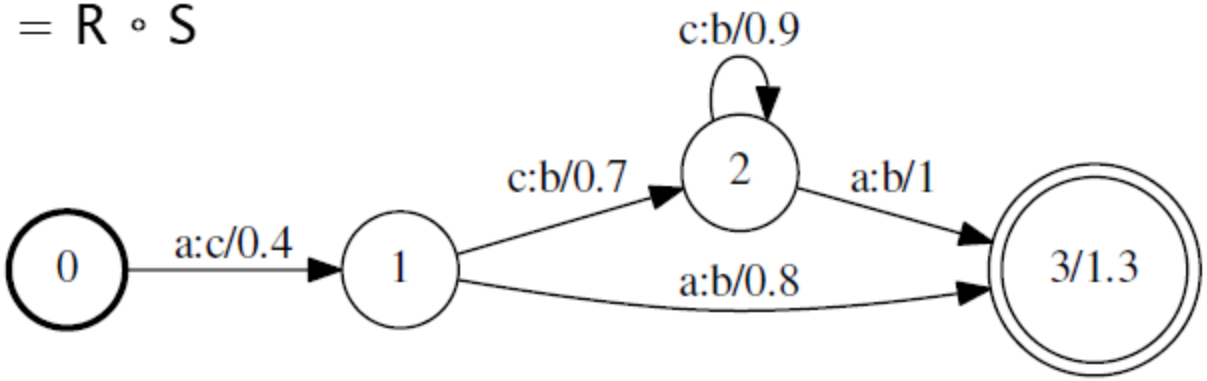
R



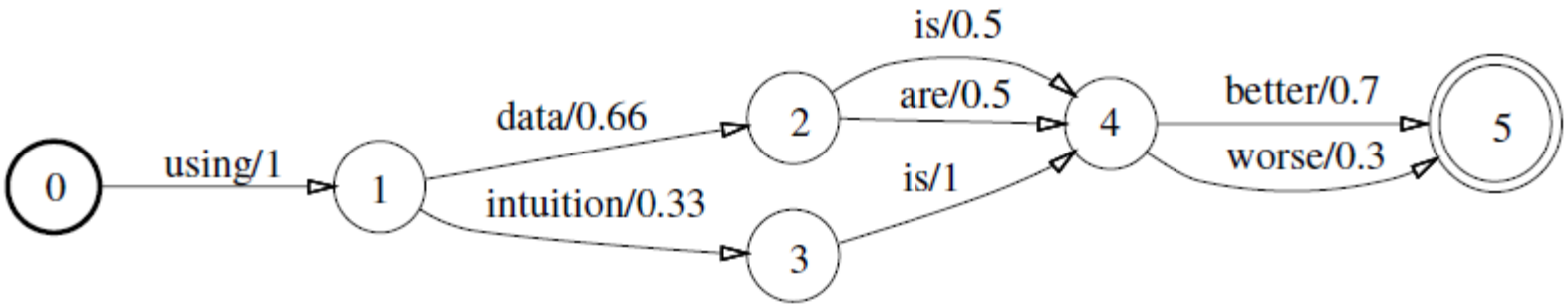
S

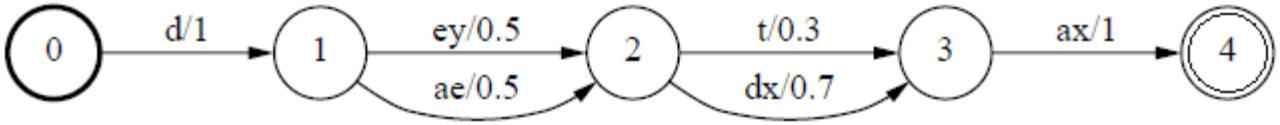


$T = R \circ S$

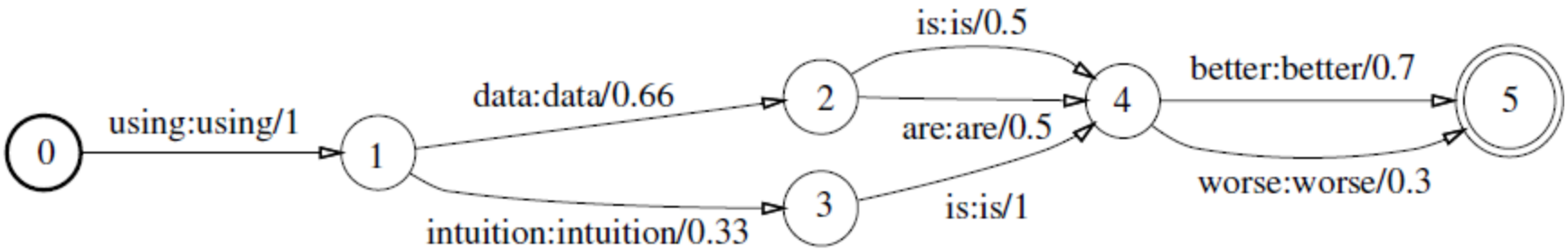


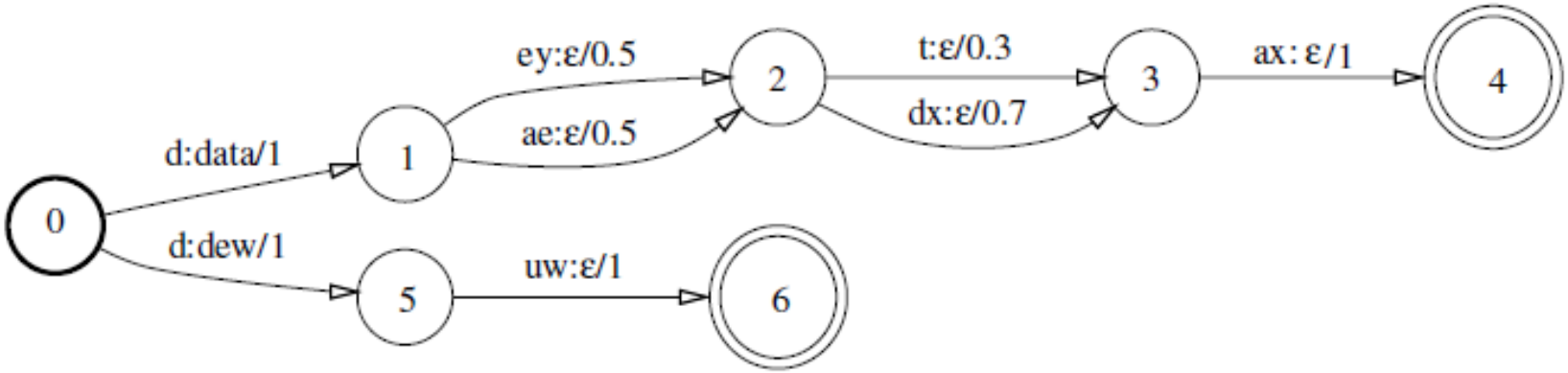
WFSA – examples





WFST – examples





FST or FSA?

- This illustrates the key advantage of a transducer over an acceptor:
  - the transducer can represent a relationship between two levels of representation, for instance between phones and words or between HMMs and context-independent phones.
- A transducer specifies a binary relation between strings:
  - two strings are in the relation when there is a path from an initial to a final state in the transducer that has the first string as the sequence of input labels along the path, and the second string as the sequence of output labels along the path
- For a weighted transducer, each string pair is also associated with a weight.

Components in ASR

- hidden Markov models (HMMs)
- context-dependency models
- pronunciation dictionaries
- statistical grammars

- They can all be represented by WFST

Decoding in ASR

- The ultimate goal of decoding in ASR is to convert the human sounds into word sequence.
- Now let's simplify the problem into "converting phoneme sequence to word sequence"

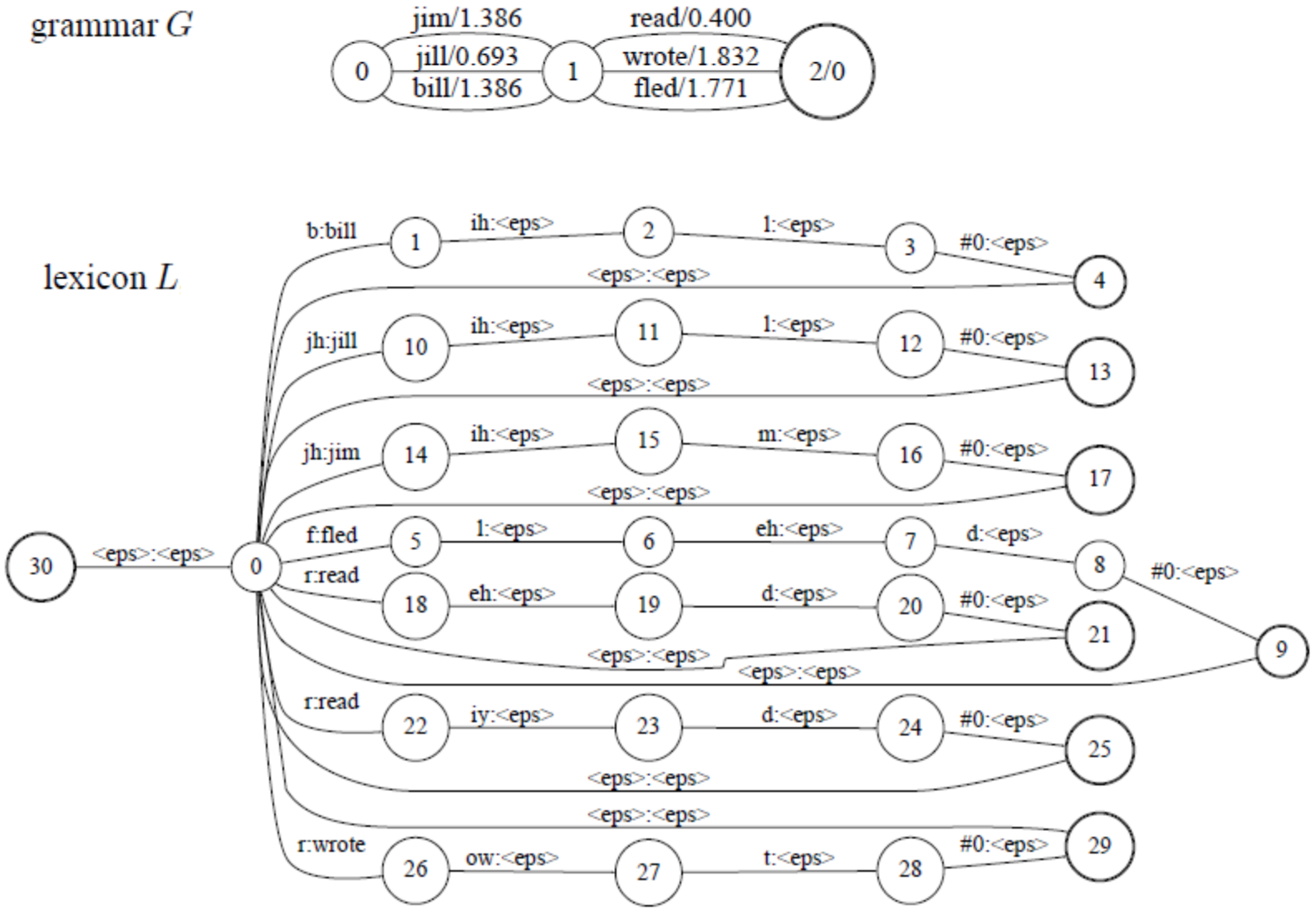
Decoding in ASR

- Given phoneme sequences:
- r eh d k ah l er
- ih n t uw d ey

- We need a systematic way.

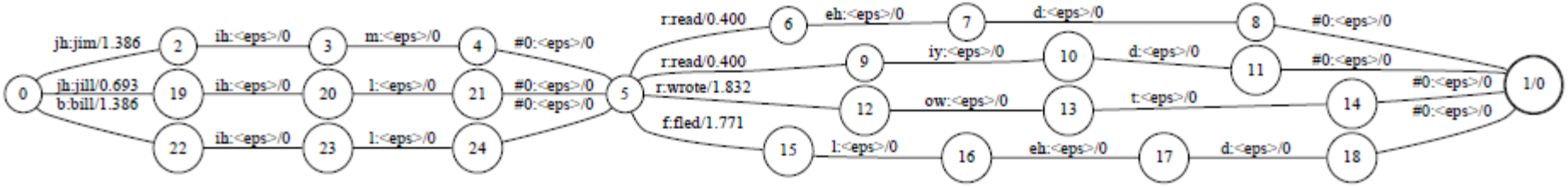
Dictionary	
RED	*r eh d*
READ	*r eh d*
COLOR	k ah l er
IN	*ih n*
INTO	ih n t uw
DAY	*d ey*
TODAY	*t uw d ey*

Composition of lexicon and grammar



Composition of lexicon and grammar

$L \circ G$



Composition of differents level in ASR

- Composition is the transducer operation for combining different levels of representation.
- For instance, a pronunciation lexicon can be composed with a word-level grammar to produce a phone-to-word transducer whose word sequences are restricted to the grammar.

Overall recipe in Kaldi

- Create a HCLG graph
- Composition of:
  - H = HMM structure
  - C = phonetic context dependency
  - L = lexicon
  - G = grammar (or language model)
- Optimizations such as determinization and minimization are done to improve decoding performance.

Determinization

- In a *deterministic automaton*, each state has at most one transition with any given input label and there are no input ϵ-labels.

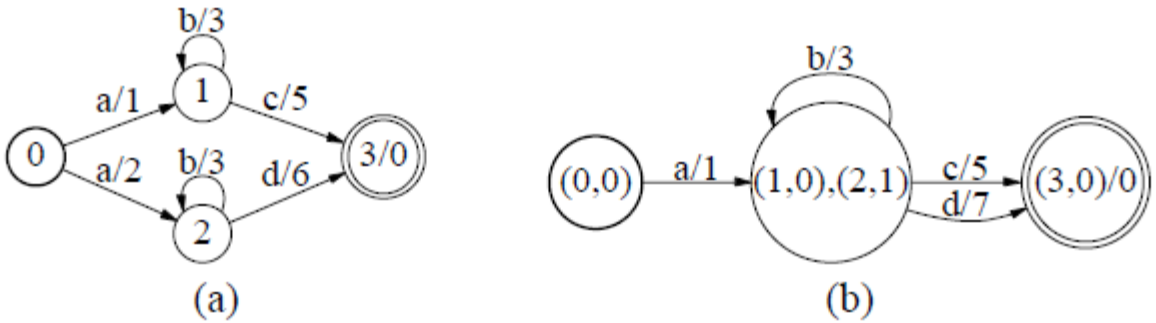


Figure 4: Determinization of weighted automata. (a) Weighted automaton over the tropical semiring A. (b) Equivalent weighted automaton B obtained by determinization of A.

Minimization

- Minimize #states and #transitions to save both space and time

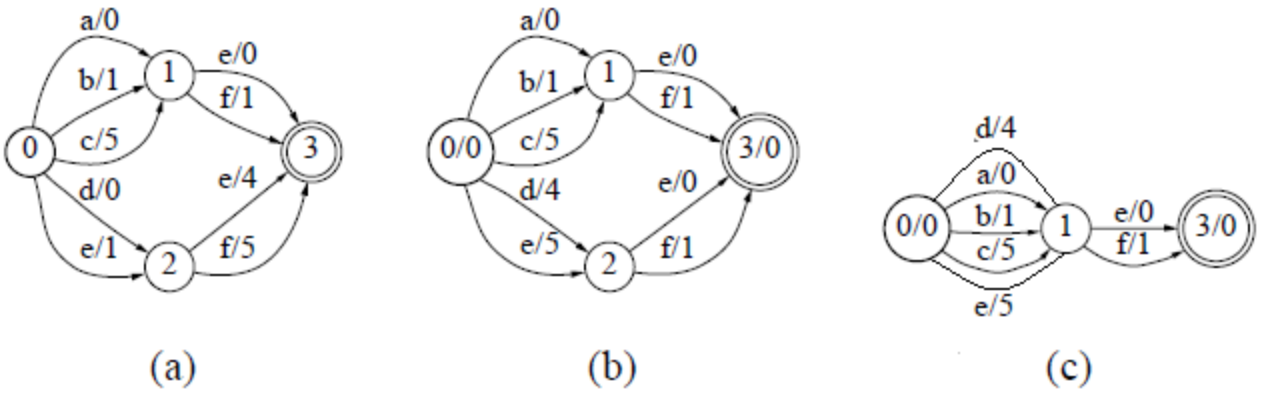


Figure 5: Weight pushing and minimization. (a) Deterministic weighted automaton A. (b) Equivalent weighted automaton B obtained by weight pushing in the tropical semiring. (c) Minimal weighted automaton equivalent to A.

What we have learnt so far

- Acoustic model, either by GMM or DNN
- Language model
- Lexicon (dictionary)
- FST

- How to bring them together?

What is the nature of ASR?

- Given a model and the audio signal, we want to know the most likely word sequence.

$$\hat{W} = \arg \max_{W \in \mathcal{V}*} p(W|X)$$

where W is the word sequence and X is the audio feature sequence.

- We need a way to obtain P(W|X), or you can say, a good way to relate X and W.

Factorize the whole model

$$\hat{W} = \arg\max_{W \in \mathcal{V}*} p(W|X)$$

$$= \arg\max_{W \in \mathcal{V}*} p(X|W)\,p(W)$$

$$= \arg\max_{W \in \mathcal{V}*} \sum_{S} p(X|S,W)p(S|W)p(W)$$

$$\approx \arg\max_{W \in \mathcal{V}*} \sum_{S} p(X|S)p(S|W)p(W)$$

Acoustic model

Lexicon

Language model

Simplify the acoustic model

- The acoustic model is factorized by using a probabilistic chain rule and conditional independence assumption.

$$p(X|S) = \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{x}_1, , \cdots, \mathbf{x}_{t-1}, S)$$
$$\approx \prod_{t=1}^{T} p(\mathbf{x}_t|s_t) \propto \prod_{t=1}^{T} \frac{p(s_t|\mathbf{x}_t)}{p(s_t)}$$

- The conditional independence assumption is too strong as the x_t in X are closely related due to coarticulation.

Simplify the lexicon

- The lexicon constraints the possible phone transitions.

$$p(S|W) = \prod_{t=1}^{T} p(s_t|s_1, \ldots, s_{t-1}, W)$$
$$\approx \prod_{t=1}^{T} p(s_t|s_{t-1}, W)$$

Simplify the language model

- The language model is factorized by using a probabilistic chain rule and conditional independence assumption as an m-gram.

$$
\begin{aligned}
p(W) = \; & p(w_1, \ldots, w_{n-1}, w_n) \\
& = \prod_{n=1}^{N} p(w_n | w_1, \ldots, w_{n-1}) \\
& \approx \prod_{n=1}^{N} p(w_n | w_{n-m-1}, \ldots, w_{n-1})
\end{aligned}
$$

Evaluation

- With all the necessary components, we can compute P(W|X) with a particular pair of W and X. (Evaluation)
- However, our problem is to look for the best W according to

$$\hat{W} \approx \arg \max_{W \in \mathcal{V}^*} \sum_S p(X|S)p(S|W)p(W)$$

- Naively speaking, if there is only a finite combination of W, it can be solved by evaluation.

Decoding

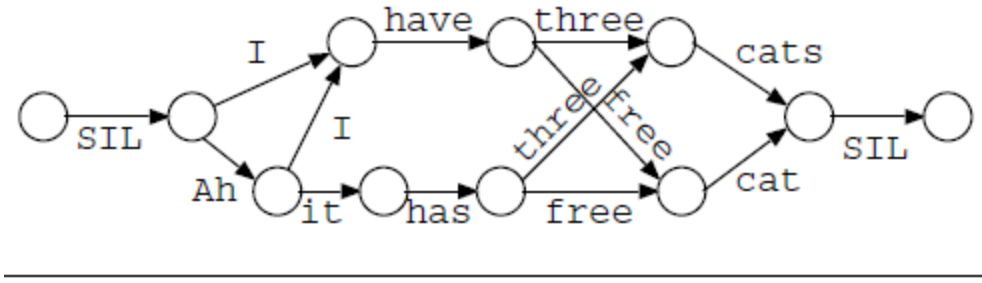
- Decoding is the process of finding the best path.
- After preparing the graph, we can use Viterbi algorithm to find the best path.

Viterbi with beam-pruning

- It is very time consuming to run a standard Viterbi on a large graph.
- Beam-pruning means removing states with score worse than the leading score. (best-score minus beam)
- For reasonable beam values, the accuracy will not be hurt too much.

Word lattice

- Instead of finding the most likely word sequence, sometimes we prefer multiple hypotheses.
  - E.g. for later language model rescoring
- Multiple hypotheses are usually represented as a word lattice or an *N*-best list.
  - A word lattice is generally more effective in representation



(a) Word lattice

```
SIL I have three cats SIL
SIL I have free cat SIL
SIL Ah it has three cats SIL
SIL Ah I have three cats SIL
                :
```

(b) *N*-best list

Readings

- Takaaki Hori, Speech Recognition Algorithms Using Weighted Finite State Transducers
- Moh97
  - Mehryar Mohri. Finite-state transducers in language and speech processing. *Computational linguistics*, 23(2):269–311, 1997.
  - https://cs.nyu.edu/~mohri/pub/cl1.pdf.
- MPR02
  - Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.
  - https://cs.nyu.edu/~mohri/postscript/csl01.pdf
- MPR08
  - Mehryar Mohri, Fernando Pereira, and Michael Riley. Speech recognition with weighted finite-state transducers. In *Springer Handbook of Speech Processing*, pages 559–584. Springer, 2008.
  - https://wiki.eecs.yorku.ca/course_archive/2011-12/W/6328/_media/wfst-lvcsr.pdf