ASR

第三课 语言模型与解码对齐

第三课 语言模型与解码对齐

- 知识点1：解码与对齐（DECODING, ALIGNMENT）：从孤立词识别到连接词/词序列之CONNECTED WORD RECOGNITION与TIME ALIGNMENT
- 知识点2： WFST介绍、WFST基本操作COMPOSITION/DETERMINISATION/MINIMISATION
- 知识点3： WFST在ASR中的应用：HCLG、基于WFST的BEAM SEARCH

REFERENCES

- HTTP://WWW.INF.ED.AC.UK/TEACHING/COURSES/ASR/2018-19/ASR11-WFST.PDF
- HTTPS://WWW.DROPBOX.COM/S/PVJODR3QDBMMVK5/LECTURE_1__INTRODUCTION_TO_FINITE_AUTOMATON.PDF?DL=0
- HTTP://DANIELPOVEY.COM/FILES/LECTURE4.PDF

Fundamental Equation of Statistical Speech Recognition

If $\mathbf{X}$ is the sequence of acoustic feature vectors (observations) and $\mathbf{W}$ denotes a word sequence, the most likely word sequence $\mathbf{W}^*$ is given by

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} P(\mathbf{W}|\mathbf{X})$$

Applying Bayes' Theorem:

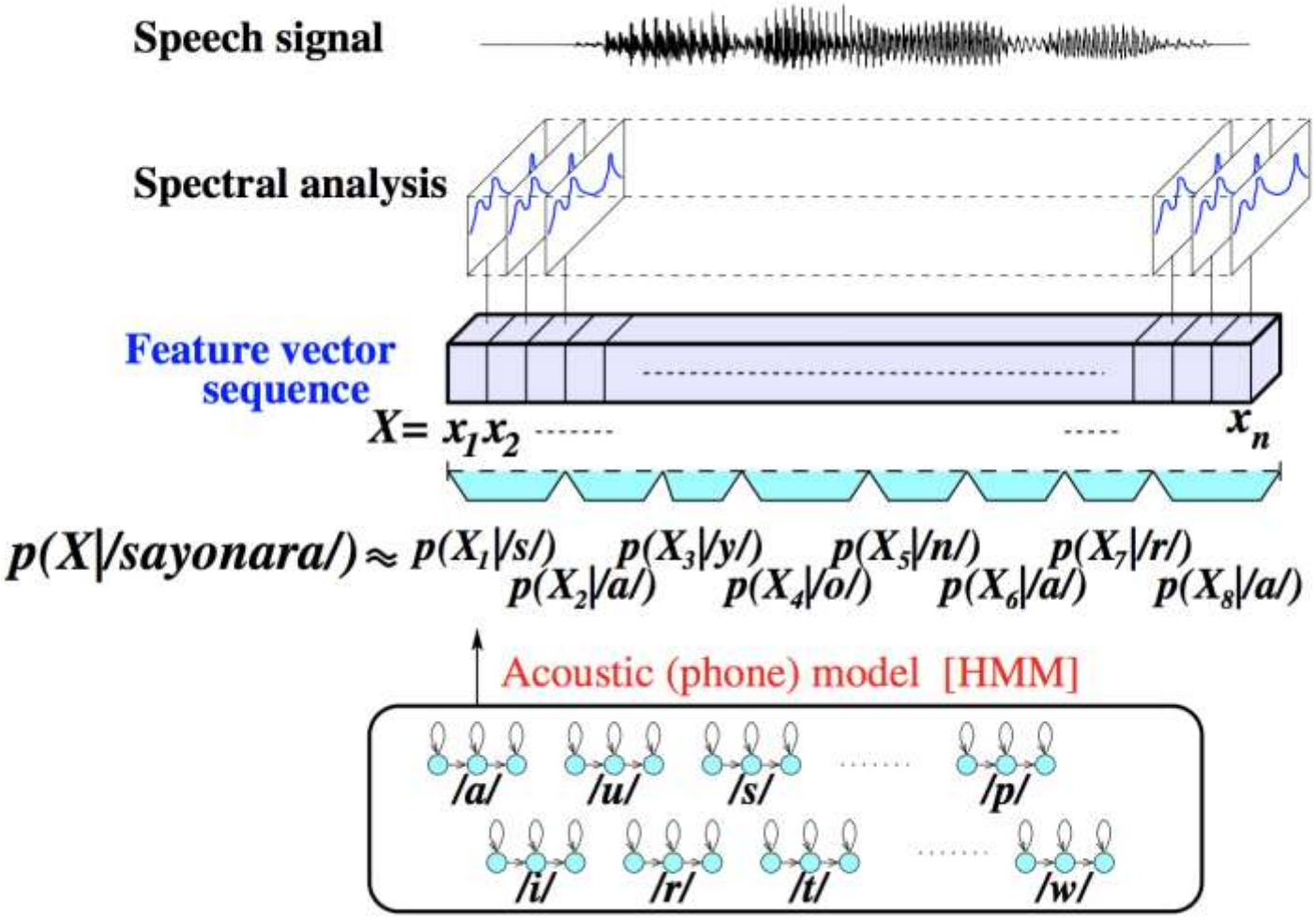
$$P(\mathbf{W}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{W})\,P(\mathbf{W})}{p(\mathbf{X})}$$

$$\propto p(\mathbf{X}|\mathbf{W})\,P(\mathbf{W})$$

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} \underbrace{p(\mathbf{X}|\mathbf{W})}_{\text{Acoustic model}} \underbrace{P(\mathbf{W})}_{\text{Language model}}$$

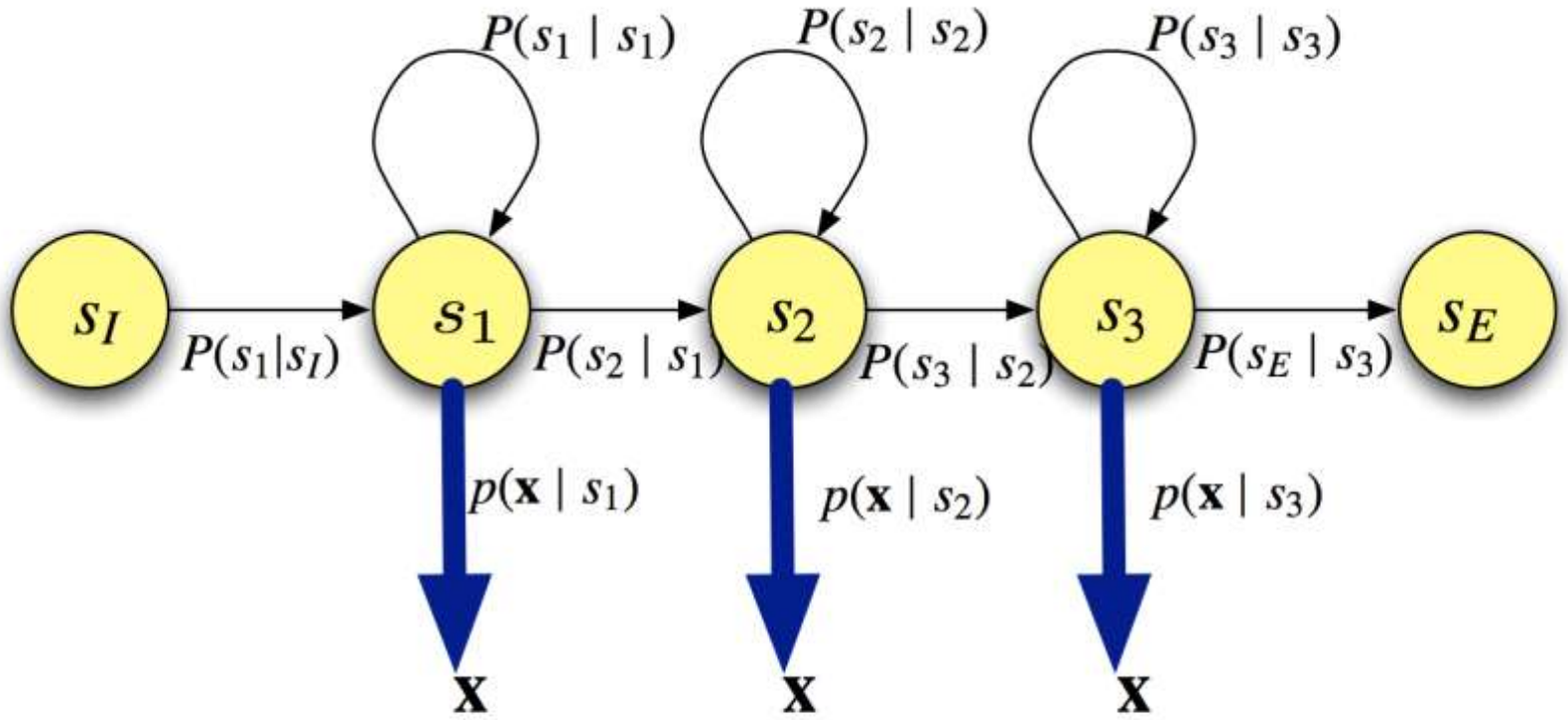
NB: $\mathbf{X}$ is used hereafter to denote the output feature vectors from the signal analysis module rather than DFT spectrum.

Calculation of $p(X|W)$



NB: some conditional independency is assumed here.

Acoustic Model: Continuous Density HMM



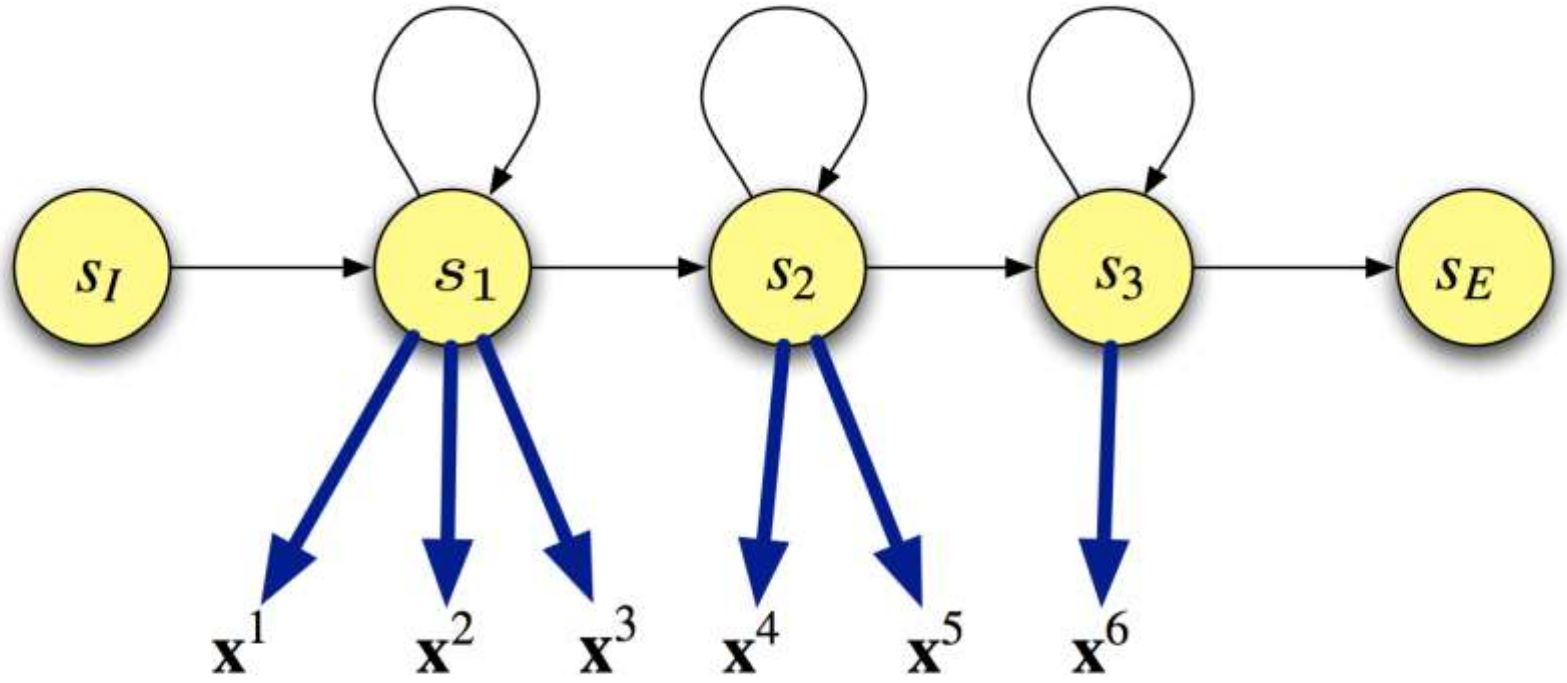
Probabilistic finite state automaton

Parameters $\boldsymbol{\lambda}$:

- Transition probabilities: $a_{kj} = P(S=j \mid S=k)$
- Output probability density function: $b_j(\mathbf{x}) = p(\mathbf{x} \mid S=j)$

NB: Some textbooks use Q or q to denote the state variable S.
$\mathbf{x}$ corresponds to $\mathbf{o}_t$ in Lecture slides 02.

Acoustic Model: Continuous Density HMM



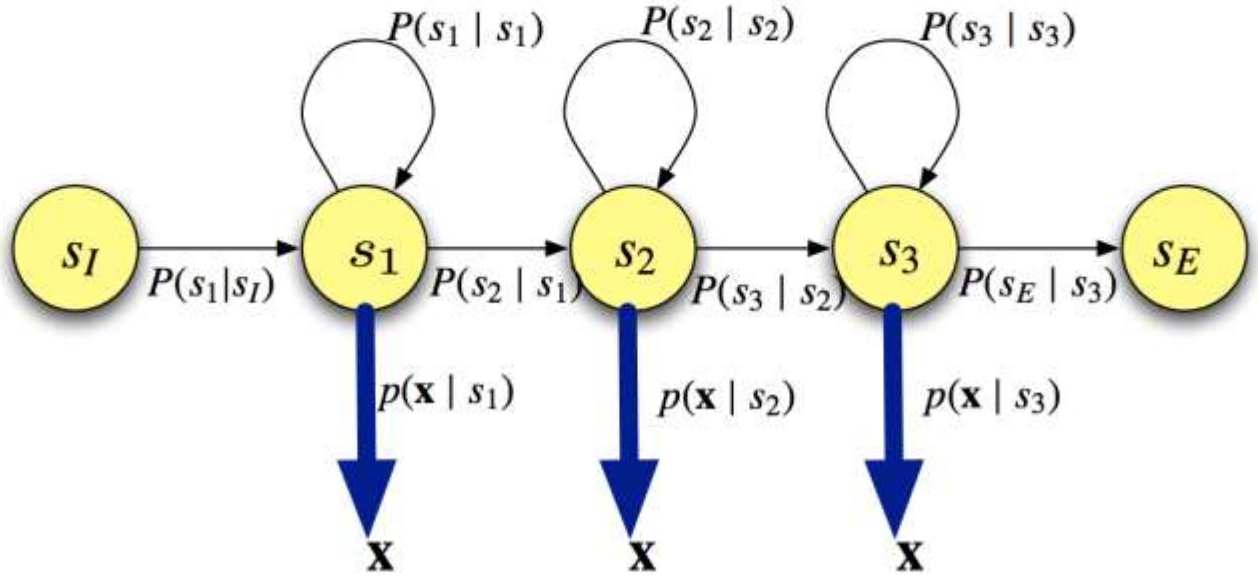
Probabilistic finite state automaton

Parameters $\boldsymbol{\lambda}$:

- Transition probabilities: $a_{kj} = P(S=j \mid S=k)$
- Output probability density function: $b_j(\mathbf{x}) = p(\mathbf{x} \mid S=j)$

NB: Some textbooks use Q or q to denote the state variable S.
$\mathbf{x}$ corresponds to $\mathbf{o}_t$ in Lecture slides 02.

Output distribution



- Single multivariate Gaussian with mean $\boldsymbol{\mu}_j$, covariance matrix $\boldsymbol{\Sigma}_j$:

$$b_j(\mathbf{x}) = p(\mathbf{x}|S=j) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

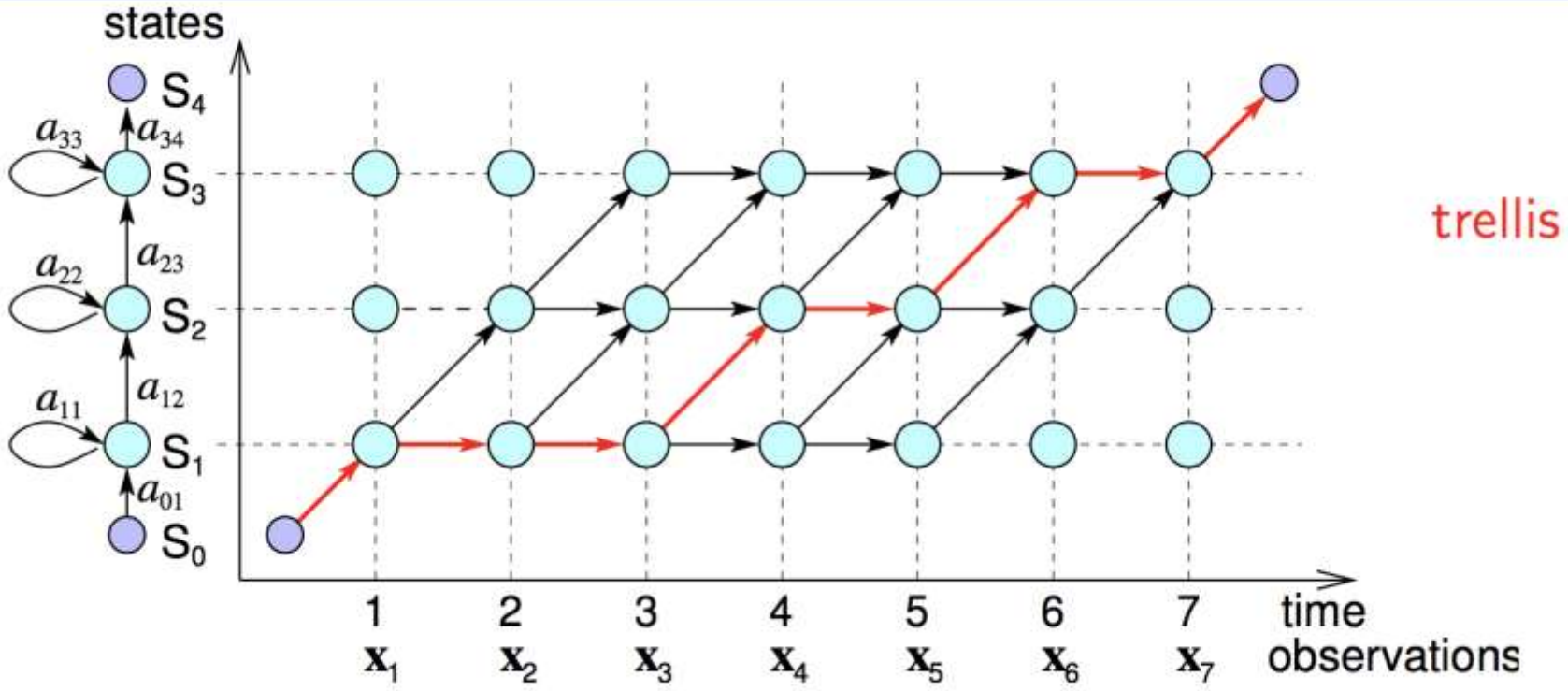
- M-component Gaussian mixture model:

$$b_j(\mathbf{x}) = p(\mathbf{x}|S=j) = \sum_{m=1}^{M} c_{jm}\, \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$$

- Neural network:

$$b_j(\mathbf{x}) \sim P(S=j|\mathbf{x}) \,/\, P(S=j)$$ NB: NN outputs posterior probabilities

1. Likelihood: how to calculate?



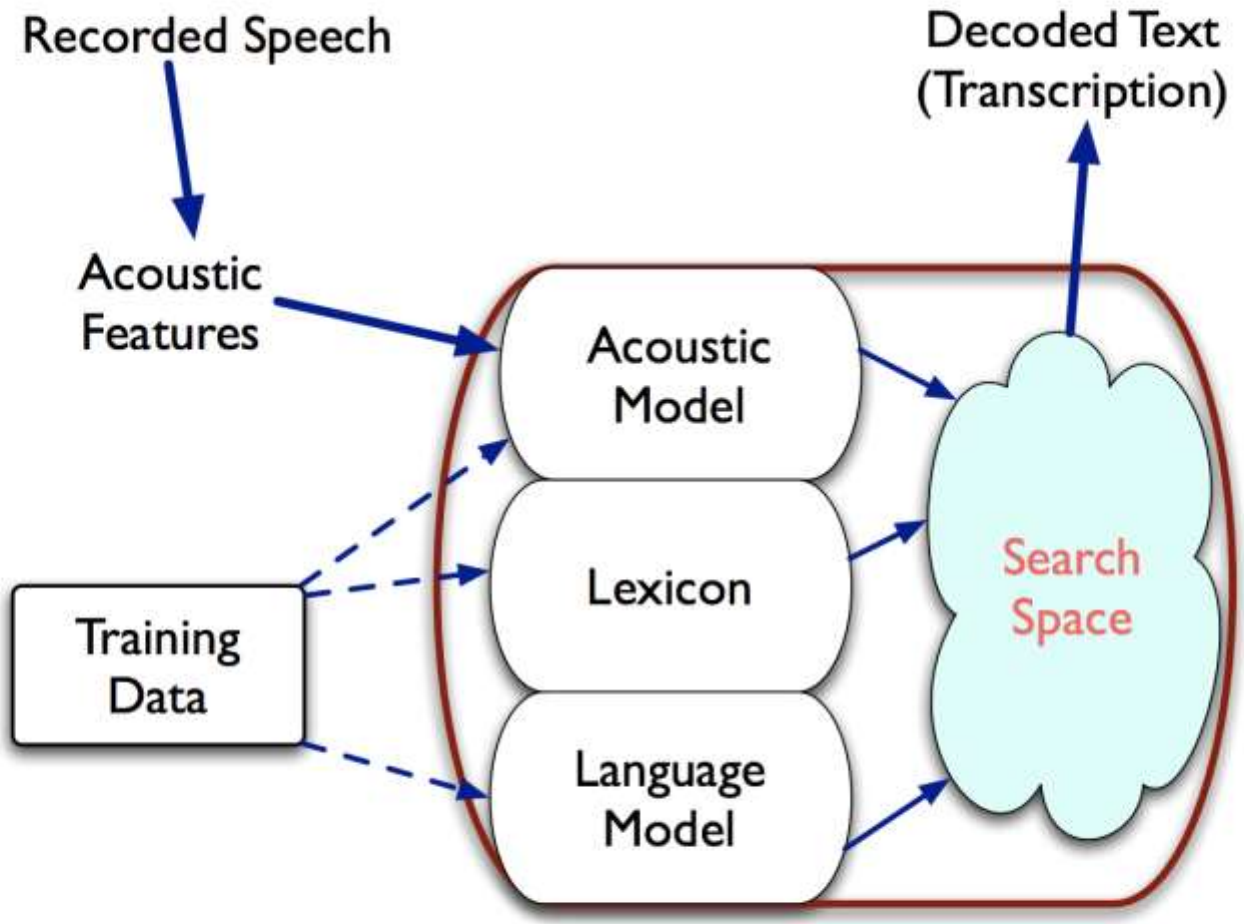
trellis

$$p(\mathbf{X}, \mathrm{path}_\ell \,|\, \boldsymbol{\lambda}) = p(\mathbf{X} \,|\, \mathrm{path}_\ell, \boldsymbol{\lambda}) P(\mathrm{path}_\ell \,|\, \boldsymbol{\lambda})$$
$$= p(\mathbf{X} \,|\, s_0 s_1 s_1 s_1 s_2 s_2 s_3 s_3 s_4, \boldsymbol{\lambda}) P(s_0 s_1 s_1 s_1 s_2 s_2 s_3 s_3 s_4 \,|\, \boldsymbol{\lambda})$$
$$= b_1(\boldsymbol{x}_1) b_1(\boldsymbol{x}_2) b_1(\boldsymbol{x}_3) b_2(\boldsymbol{x}_4) b_2(\boldsymbol{x}_5) b_3(\boldsymbol{x}_6) b_3(\boldsymbol{x}_7) a_{01} a_{11} a_{11} a_{12} a_{22} a_{23} a_{33} a_{34}$$

$$p(\mathbf{X} \,|\, \boldsymbol{\lambda}) = \sum_{\{\mathrm{path}_\ell\}} p(\mathbf{X}, \mathrm{path}_\ell \,|\, \boldsymbol{\lambda}) \simeq \max_{\mathrm{path}_\ell} p(\mathbf{X}, \mathrm{path}_\ell \,|\, \boldsymbol{\lambda})$$

forward(backward) algorithm Viterbi algorithm

HMM Speech Recognition



The Search Problem in ASR

- Find the most probable word sequence $\hat{W} = w_1, w_2, \ldots, w_M$ given the acoustic observations $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$:

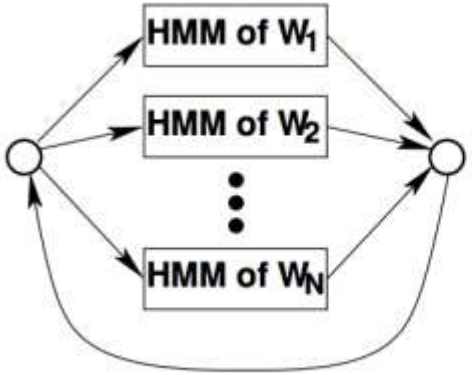
$$\hat{W} = \arg\max_W P(W|\mathbf{X})$$

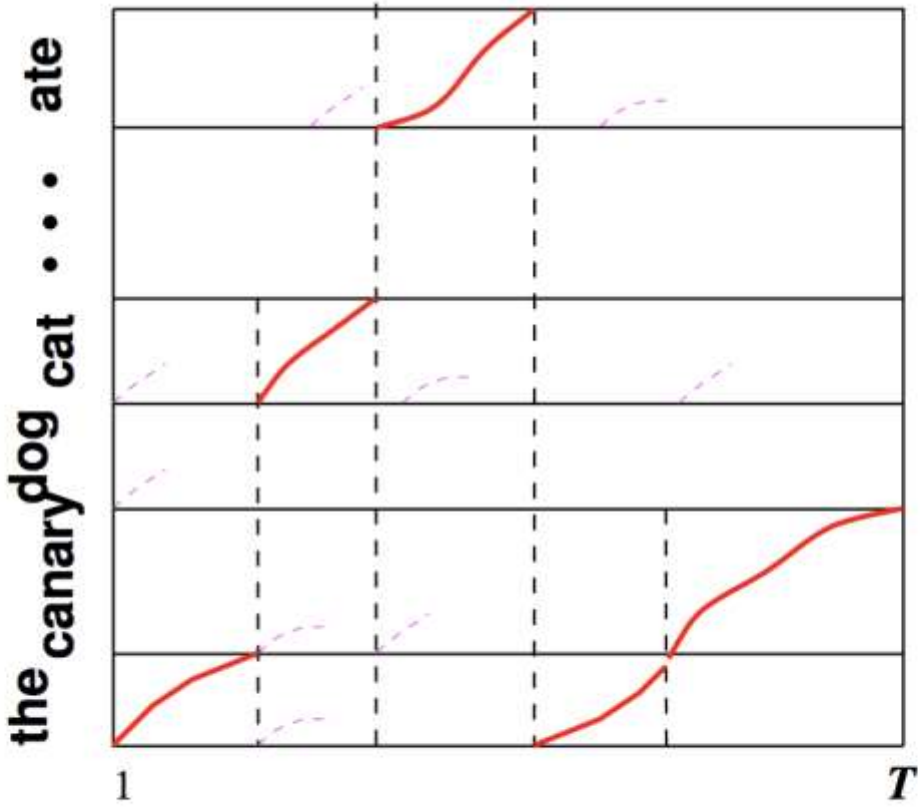
$$= \arg\max_W \underbrace{p(\mathbf{X} \mid W)}_{\text{acoustic model}} \underbrace{P(W)}_{\text{language model}}$$

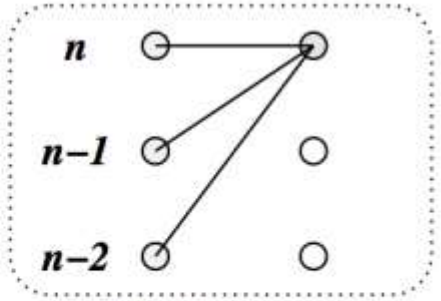
- Words are composed of state sequences so this problem corresponds to finding the most probable allowable state sequence (given the constraints of pronunciation lexicon and language model) - **Viterbi decoding**
- In a large vocabulary task evaluating all possible word sequences in infeasible (even using an efficient exact algorithm)
  - Reduce the size of the search space through pruning unlikely hypotheses
  - Eliminate repeated computations

Connected Word Recognition

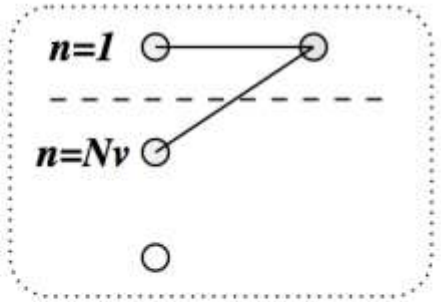
- The number of words in the utterance is not known
- Word boundaries are not known: V words may potentially start at each frame







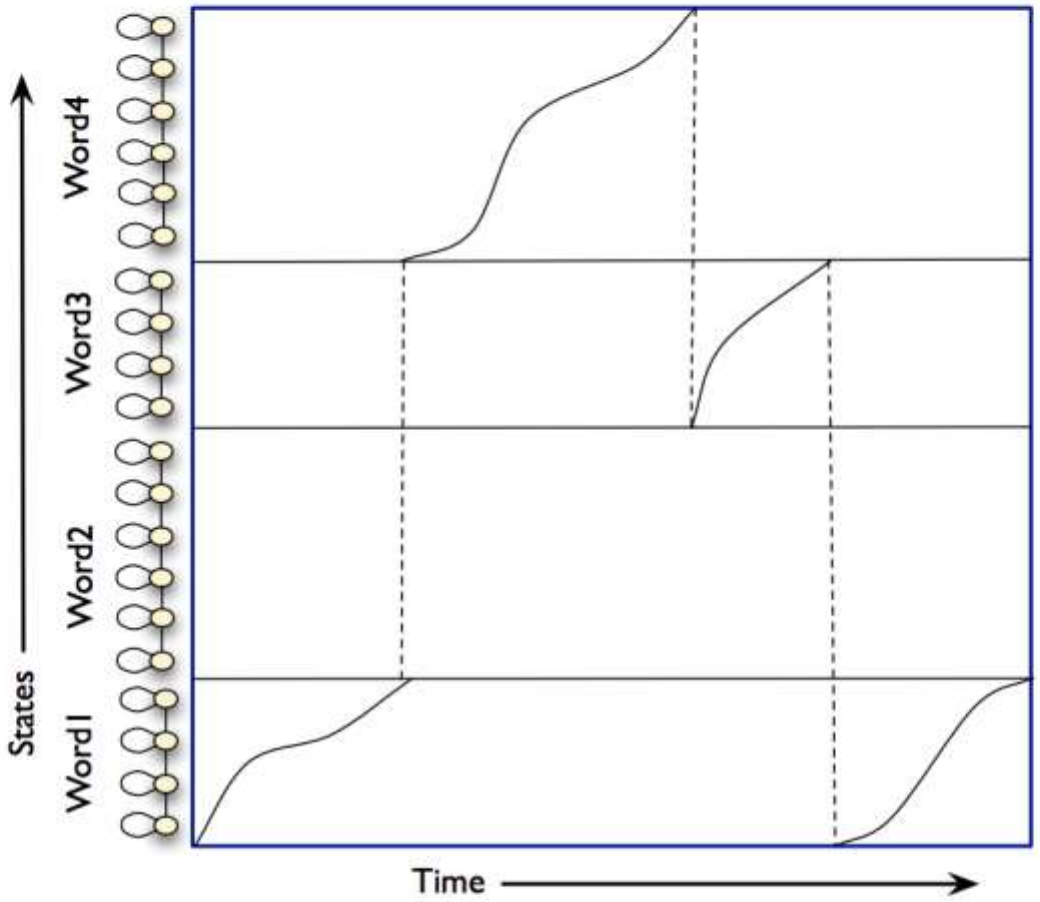
(a) intra word



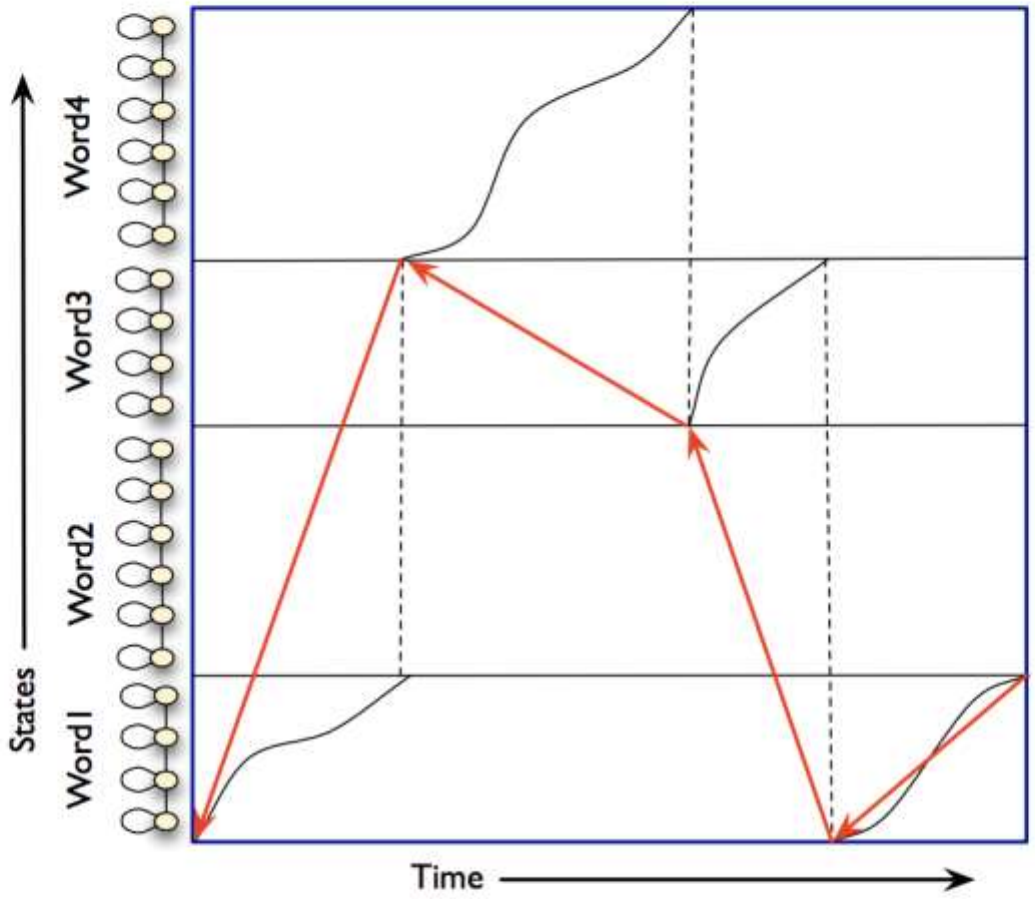
(b) intra/inter word

speech: "the cat ate the canary"

Time Alignment Path

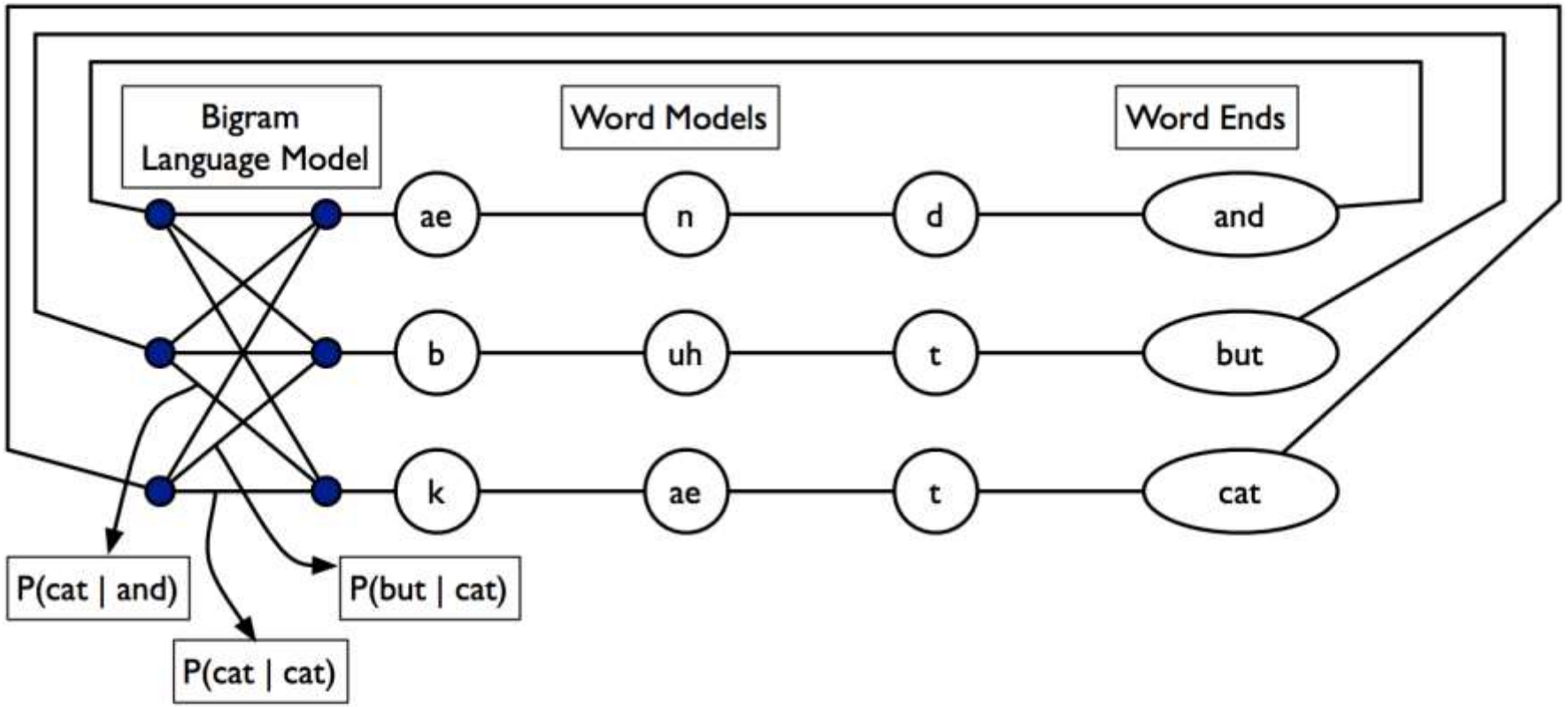


Backtrace to Obtain Word Sequence



- Backpointer array keeps track of word sequence for a path:
  backpointer[word][wordStartFrame] = (prevWord, prevWordStartFrame)
- Backtrace through backpointer array to obtain the word sequence for a path

Incorporating a bigram language model



Trigram or longer span models require a word history.

Computational Issues

- Viterbi decoding performs an exact search in an efficient manner
- Exact search is not possible for large vocabulary tasks
  - Cross-word triphones need to be handled carefully since the acoustic score of a word-final phone depends on the initial phone of the next word
  - Long-span language models (eg trigrams) greatly increase the size of the search space
- Solutions:
  - Beam search (prune low probability hypotheses)
  - Dynamic search structures
  - Multipass search ($\rightarrow$ two-stage decoding)
  - Best-first search ($\rightarrow$ stack decoding / A* search)
- An alternative approach: Weighted Finite State Transducers (WFST)

Weighted Finite State Transducers

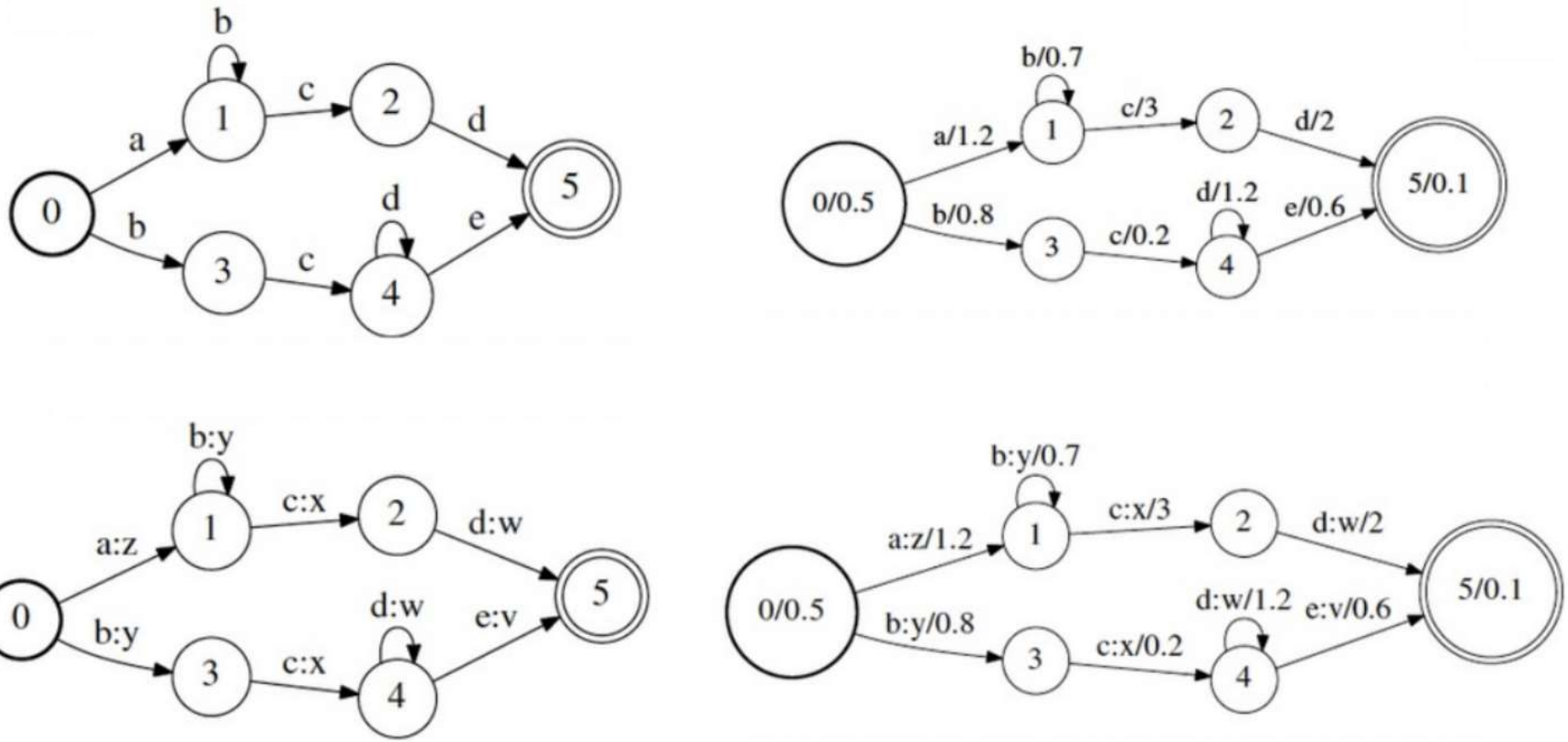
- Used by Kaldi
- Weighted finite state automaton that transduces an input sequence to an output sequence (Mohri et al 2008)
- States connected by transitions. Each transition has
  - input label
  - output label
  - weight

第三课 语言模型与解码对齐

- 知识点1：解码与对齐（DECODING, ALIGNMENT）：从孤立词识别到连接词/词序列之CONNECTED WORD RECOGNITION与TIME ALIGNMENT
- 知识点2： WFST介绍、WFST基本操作COMPOSITION/DETERMINISATION/MINIMISATION
- 知识点3： WFST在ASR中的应用：HCLG、基于WFST的BEAM SEARCH

Informally, it is a collection *of a finite set of states and state transitions, where each transition has at least one label.*

Figure: Some examples of Finite Automata



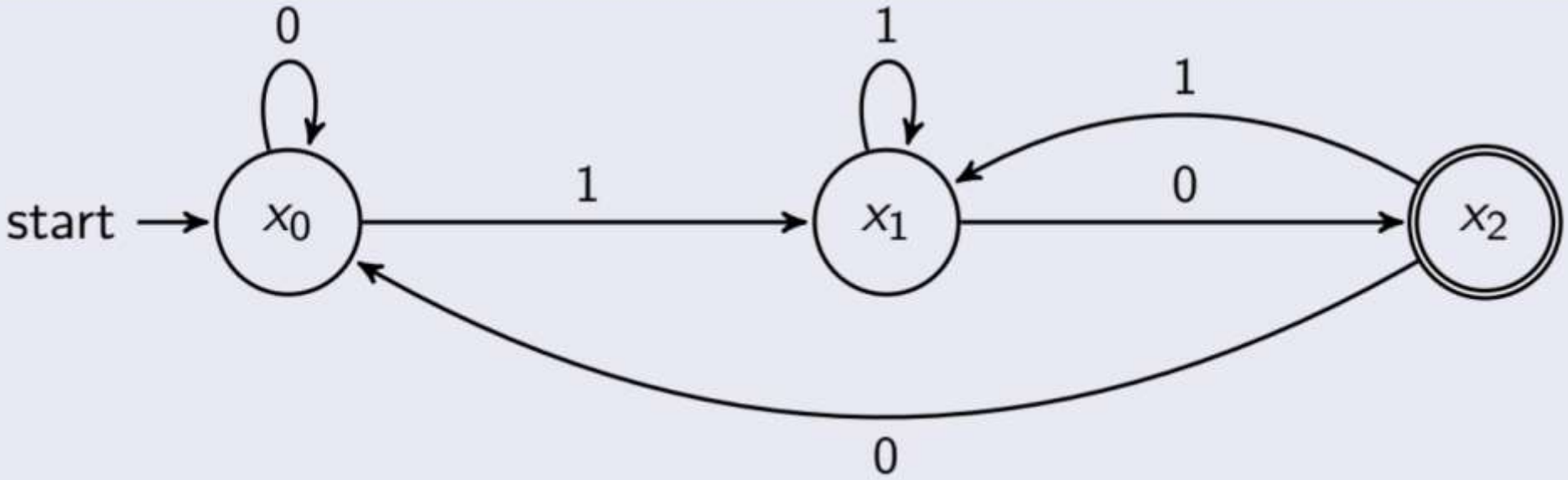
Summary on Types of FA

Table 3.1: Types of finite automaton

type	input	output	weight	mapping
Finite-state Acceptor (FSA)	✓			$\Sigma \rightarrow \{0, 1\}$
Finite-state Transducer (FST)	✓	✓		$\Sigma \rightarrow 2^{\Delta}$
Weighted FSA (WFSA)	✓		✓	$\Sigma \rightarrow \mathbb{K}$
Weighted FST (WFST)	✓	✓	✓	$\Sigma \rightarrow 2^{\Delta} \times \mathbb{K}$

Applications of FA

Finding all occurrences of pattern "10" in a binary string



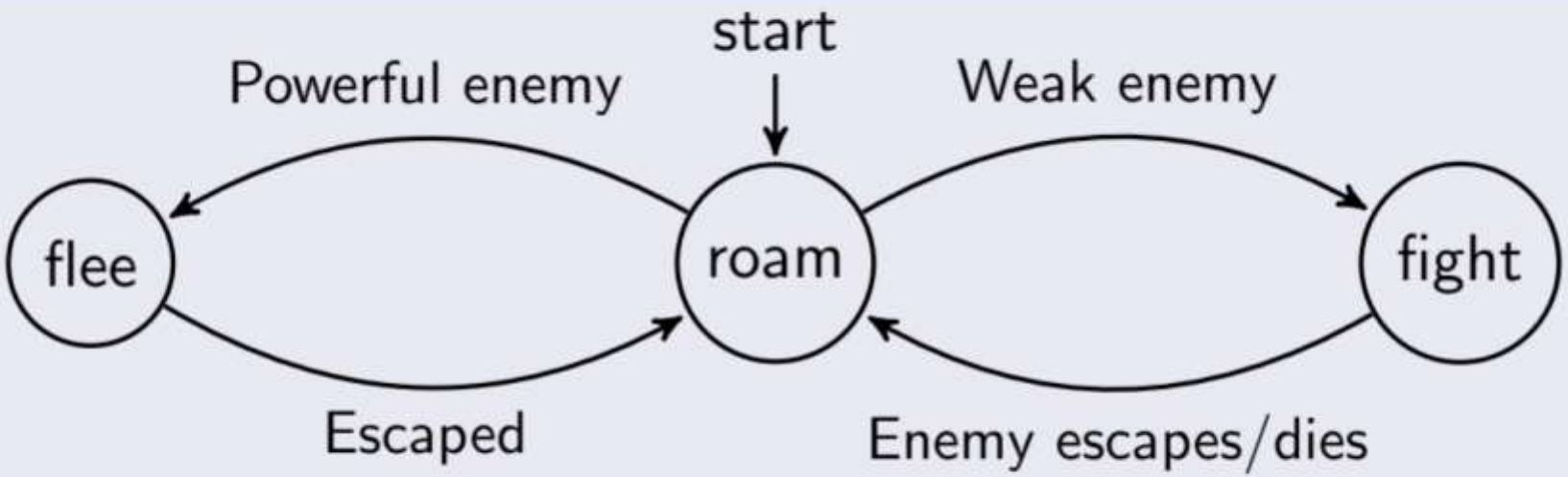
Consider 3 binary strings:

- 001001 ✓
- 000111 ✗
- 101010 ✓

Applications of FA

- A Bot is a character in a game that is controlled by the computer (Not NPC!).

Coding Bot behaviours in games



- And many other applications as well.

Finite Automata (*Definition*)

5 Tuple written as (Q, A, E, I, F)

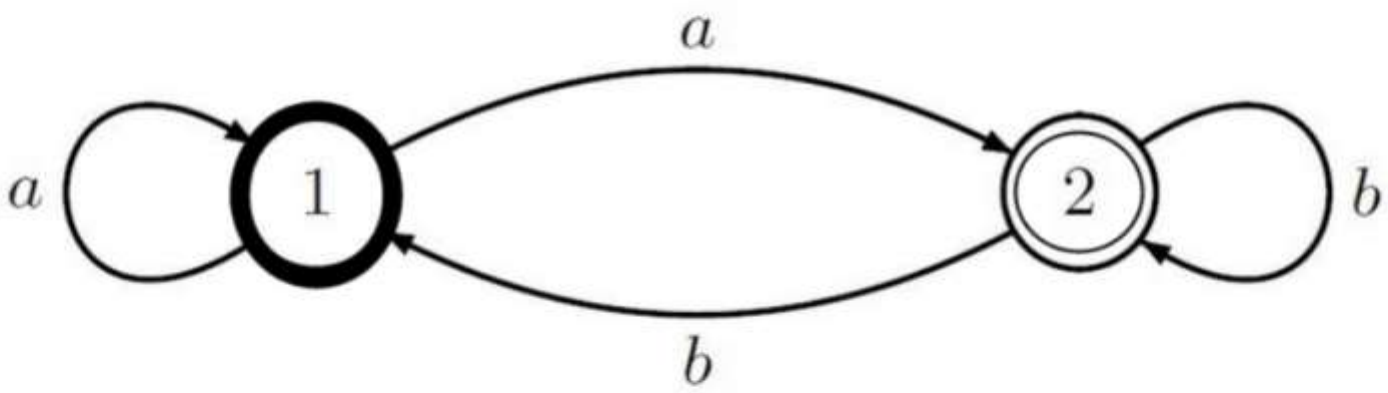
Where

- Q: Finite set of states
- A: Alphabet, or a formal language
- E: Finite set of state transitions
- I: Set of Initial states
- F: Set of Final states

Worked Example

Let $C = (Q, A, E, I, F)$
Where $Q = \{1, 2\}, I = \{1\}, F = \{2\}, A = \{a, b\}$
and $E = \{(1, a, 1), (1, a, 2), (2, b, 1), (2, b, 2)\}$

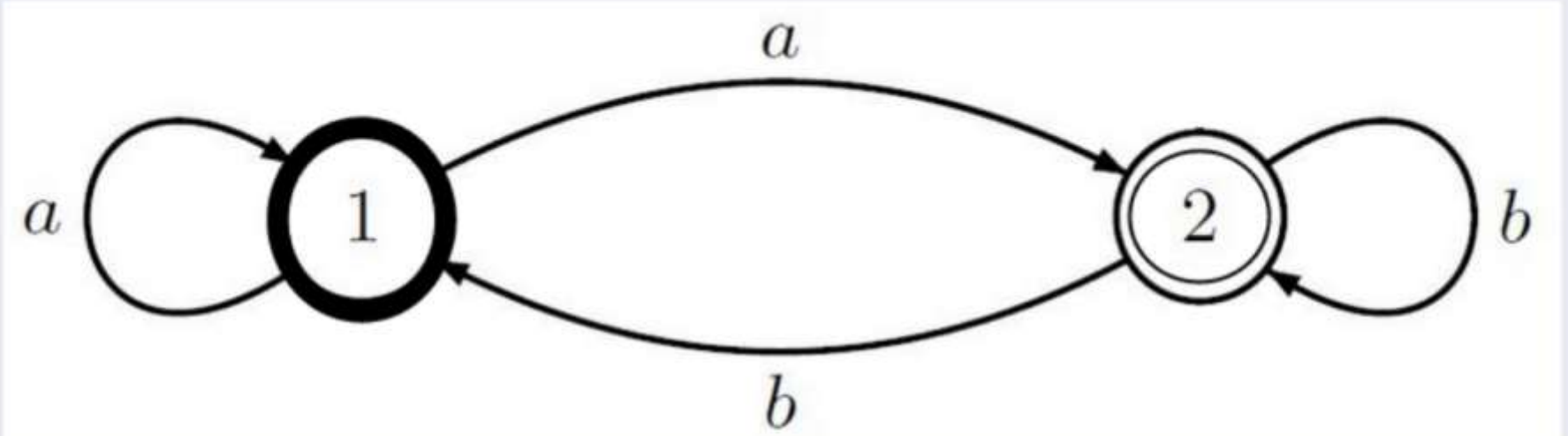
Solution:



Intuition of Path

- Note that this is a series of directed edges, $\pi = e_1 \cdots e_n$

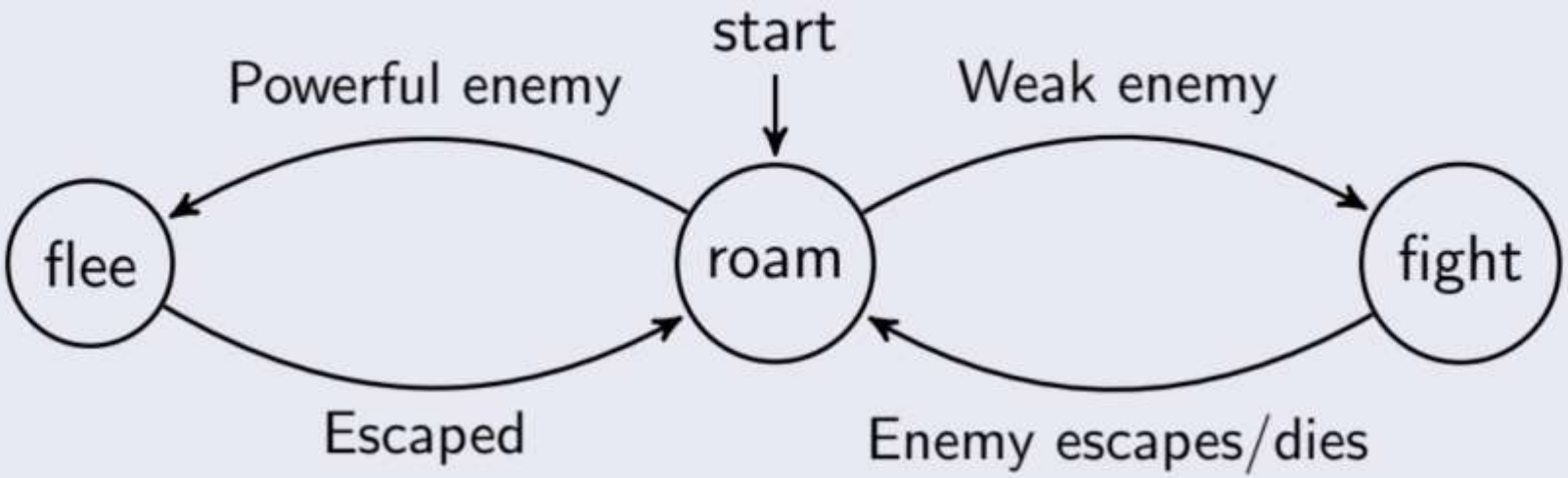
An Example



- $\pi_1 = (1, a, 1), (1, a, 2), (2, b, 2), (2, b, 1), (1, a, 2), (2, b, 2)$ is a path

Intuition of Path

Another Example: (Bot Behaviour)



- $\pi_1 = (\text{roam}, \text{Powerful Enemy}, \text{flee}), (\text{flee}, \text{Escaped}, \text{roam})$ is a path

Definition of Paths in FA

Given a Path: $\pi = e_1 \cdots e_n$

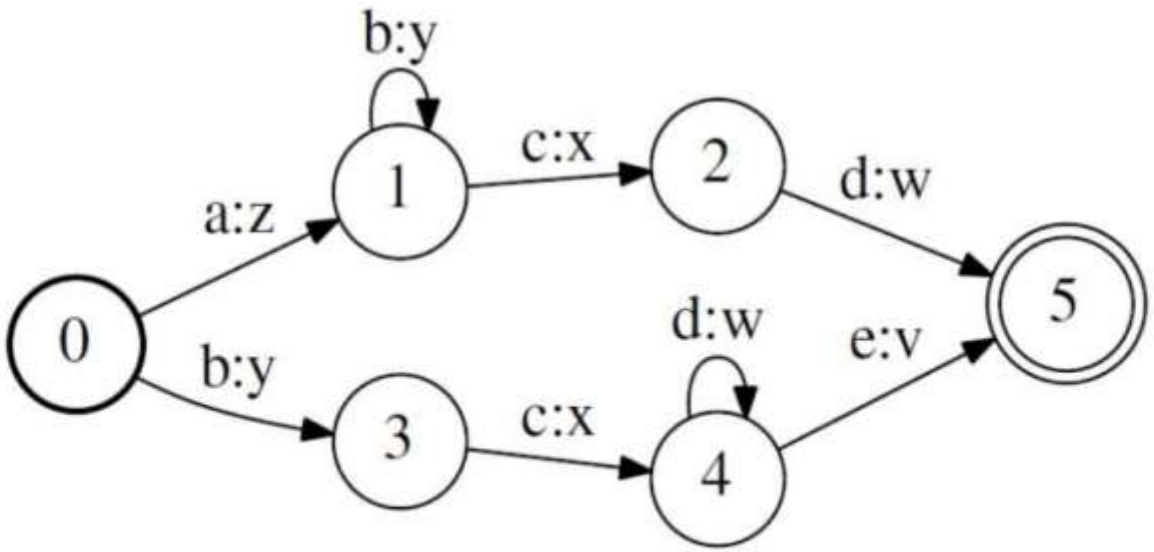
- Origin or **Previous** state: $p[\pi]$, e.g $p[\pi] = e_1$
- Destination or **Next** state: $n[\pi]$, e.g $n[\pi] = e_n$
- **Input** Label: $i(\pi)$
- **Output** Label: $o(\pi)$

Set of Paths *(Definition)*

- $P(R_1, R_2)$: Set of paths from $R_1 \subseteq Q$ to $R_2 \subseteq Q$
- $P(R_1, x, R_2)$: With Input label $x \in \Sigma$
- $P(R_1, x, y, R_2)$: With Output label $y \in \Delta$

What is $Q \times \{\Sigma \cup \varepsilon\} \times \{\Delta \cup \varepsilon\} \times Q$?

Consider a *walk*, that goes from $0 \rightarrow 1 \rightarrow 2 \rightarrow 5$



(b) Finite-State Transducer (FST)

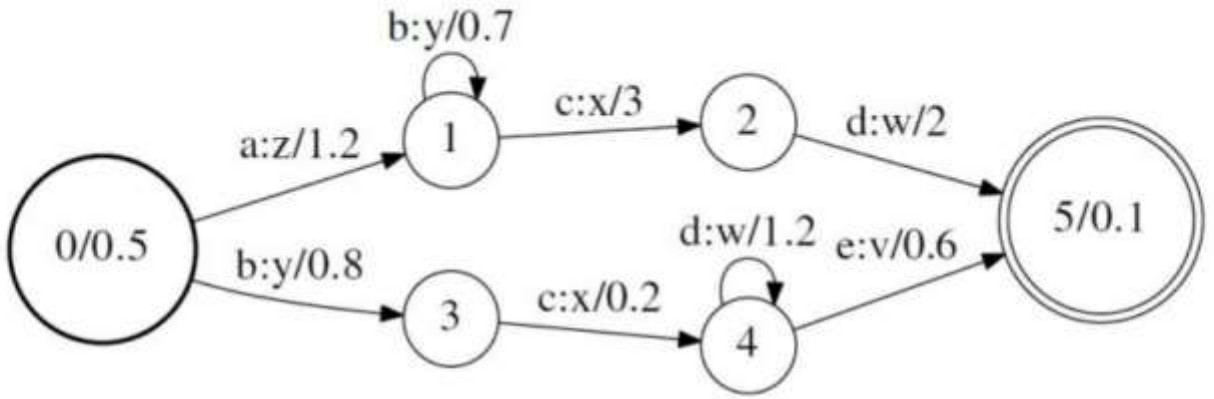
Formally, the path is written as:

- $\pi_1 = (0, a, z, 1), (1, c, x, 2), (2, d, w, 5)$

Notice each 4 tuple has a *format* of $Q \times \{\Sigma \cup \varepsilon\} \times \{\Delta \cup \varepsilon\} \times Q$?

Intuition of WFST

WFST is a *weighted* FST, each transition, initial state and final state has an additional weight associated to them.



(d) Weighted Finite-State Transducer (WFST)

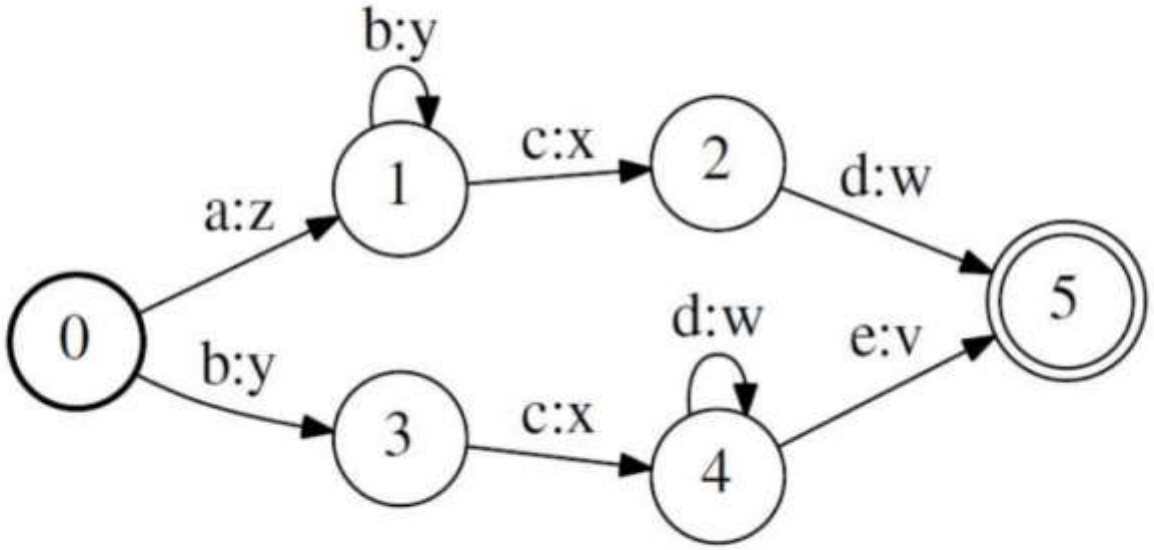
Eg: Input *abbcd*, the WFST will return *zyyxw* and the weight of the path, $0.5 \otimes 1.2 \otimes 0.7 \otimes 0.7 \otimes 3 \otimes 2 \otimes 0.1$.

Weighted Finite State Transducer (*Definition*):

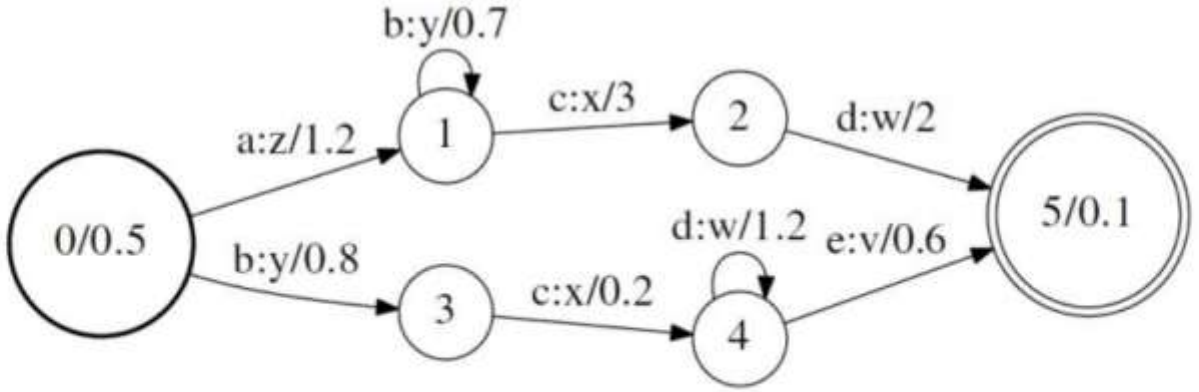
8 Tuple written as $(\Sigma, \Delta, Q, E, I, F, \lambda, \rho)$

- Σ: Set of *input* labels
- Δ: Set of *output* labels
- Q: Finite set of states
- E: Finite set of state transitions
  - $E \subseteq Q \times \{\Sigma \cup \varepsilon\} \times \{\Delta \cup \varepsilon\} \times \mathbb{K} \times Q$
- I: Set of Initial states
- F: Set of Final states
- λ: Initial weight function $(\lambda : I \to \mathbb{K})$
- ρ: Final weight function $(\rho : F \to \mathbb{K})$

Differences in FST and WFST



(b) Finite-State Transducer (FST)



(d) Weighted Finite-State Transducer (WFST)

Basic Properties of Transducers

Given $T : (\Sigma, \Delta, Q, E, I, F, \lambda, \rho)$ and $x \in \Sigma, y \in \Delta$

- Deterministic: T has no two transitions leaving the same state sharing the same input label
- Sequential: T has a unique initial state
- Functional: T has a unique y for any x
- Epsilon-Transitions: $\exists$ a state transition that can be made without any input (ε)
- Stochastic WFST: For every state, the sum of all transitions leaving that state is **EXACTLY** 1

Motivation and Importance I

- Semirings are a type of *Algebraic Structure*
- In order to manipulate WFSTs, we need to learn the concept of semirings.
- In WFST-based speech recognition, the *tropical semiring* is mainly used
- In some *optimization* steps for WFSTs, the *log semiring* is also used
- The **operations** on WFSTs will be introduced in detail in the following sections.

Motivation and Importance II

Reasons Computer Scientists are interested in semirings:

- Reduce problem complexity
- Enable **Generic Problem Solving**

Reasons Mathematicians are interested in semirings:

- Semirings are fundamentally different from fields
- New areas of research thanks to applications in CS

What are Semirings: Algebra revisited I

Base on your own knowledge, these should look familiar to you:

- $1 + (2 + 3) = (1 + 2) + 3$
- $3 + 4 = 4 + 3$
- $3 + 0 = 3$
- $1 + (-1) = 0$
  - $2 \times (3 + 4) = (2 \times 3) + (2 \times 4)$
  - $(3 + 4) \times 2 = (3 \times 2) + (4 \times 2)$
- $2 \times (3 \times 4) = (2 \times 3) \times 4$
- $2 \times 5 = 5 \times 2$
- $2 \times 1 = 2$
- $2 \times 0.5 = 1$

In pure mathematics, there is a rigorous study and extension of such algebraic structures, known as *Abstract Algebra*

What are Semirings: Algebra revisited II

Rewriting into classical algebra, given $x, y, z \in \mathbb{R}$:

- $x + (y + z) = (x + y) + z$
- $x + y = y + x$
- $x + 0 = x$
- $x + (-x) = 0$
  - $z \cdot (x + y) = (z \cdot x) + (z \cdot y)$
  - $(x + y) \cdot z = (x \cdot z) + (y \cdot z)$
- $x \cdot (y \cdot z) = (x \cdot y) \cdot z$
- $x \cdot y = y \cdot x$
- $x \cdot 1 = x$
- $x \cdot x^{-1} = 1,\ x \neq 0$

All of the *properties* each have their unique names.

What are Semirings: Algebra revisited III

Addition:

- Associativity:
  - $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
- Commutativity:
  - $x \oplus y = y \oplus x$
- Identity, $\bar{0}$: $x \oplus \bar{0} = x$
- Inverse, $-x : x \oplus (-x) = \bar{0}$

Multiplication:

- Associativity:
  - $x \otimes (y \otimes z) = (x \otimes y) \otimes z$
- Commutativity:
  - $x \otimes y = y \otimes x$
- Identity, $\bar{1}$: $x \otimes \bar{1} = x$
- Inverse, x^{-1} : $x \otimes x^{-1} = \bar{1}$

- Distributivity of Multiplication over Addition:
  - $z \otimes (x \oplus y) = (z \otimes x) \oplus (z \otimes y)$
  - $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$

The algebraic structure in the above case is known as a *FIELD*.
Recall **POLYNOMIALS**

What are Semirings: Algebra revisited IV

- Addition:
  - Associativity:
    - $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
  - Commutativity:
    - $x \oplus y = y \oplus x$
  - Identity, $\bar{0}$: $x \oplus \bar{0} = x$
  - Inverse, $-x : x \oplus (-x) = \bar{0}$
- Multiplication:
  - Associativity:
    - $x \otimes (y \otimes z) = (x \otimes y) \otimes z$
  - ~~Commutativity~~:
    -
  - Identity, $\bar{1}$: $x \otimes \bar{1} = x$
  - ~~Inverse~~
- Distributivity of Multiplication over Addition:
  - $z \otimes (x \oplus y) = (z \otimes x) + (z \otimes y)$
  - $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$

The algebraic structure in the above case is known as a *RING*.
Recall **SQUARE MATRICES**

What are Semirings: Algebra revisited V

Addition:

- Associativity:
  - $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
- Commutativity:
  - $x \oplus y = y \oplus x$
- Identity, $\bar{0}$: $x \oplus \bar{0} = x$
- ~~Inverse~~

Multiplication:

- Associativity:
  - $x \otimes (y \otimes z) = (x \otimes y) \otimes z$
- ~~Commutativity~~:
  -
- Identity, $\bar{1}$: $x \otimes \bar{1} = x$
- ~~Inverse~~

- Distributivity of Multiplication over Addition:
  - $z \otimes (x \oplus y) = (z \otimes x) + (z \otimes y)$
  - $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$

The algebraic structure above is known as a *SEMIRING*.
Recall **PROBABILITY**

Semiring (*Definition*):

5 Tuple written as $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

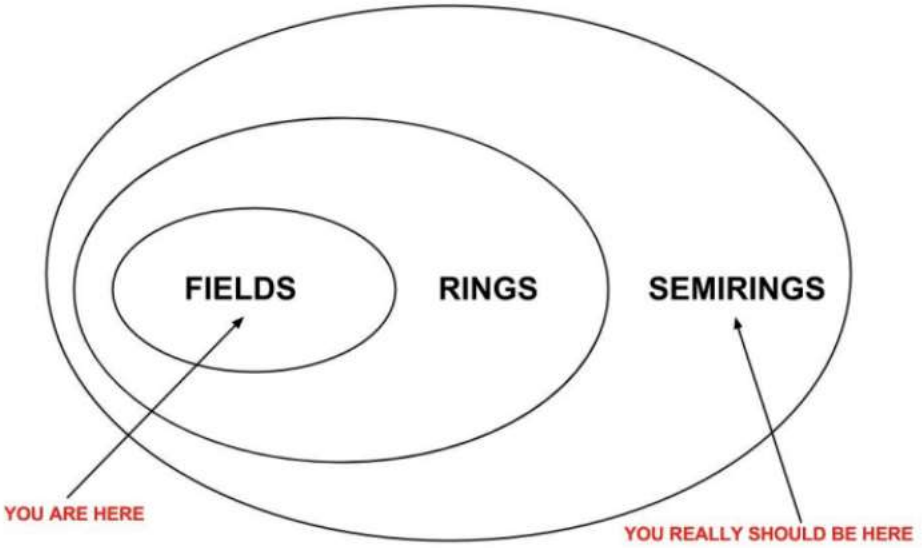
Where

- $\mathbb{K}$: A set containing some mathematical objects
- $\oplus$: Addition operation
- $\otimes$: Multiplication operation
- $\bar{0}$: Additive Identity. *Note*: $\bar{0} \in \mathbb{K}$
- $\bar{1}$: Multiplicative Identity. *Note*: $\bar{1} \in \mathbb{K}$

- $\oplus$-Addition: Compute the weight of a sequence
  - Associativity: $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
  - Commutativity: $x \oplus y = y \oplus x$
  - Additive Identity: $x \oplus \bar{0} = x$
- $\otimes$-Multiplication: Compute the weight of a path
  - Associativity: $x \otimes (y \otimes z) = (x \otimes y) \otimes z$
  - Distributivity:
    - $z \otimes (x \oplus y) = (z \otimes x) \oplus (z \otimes y)$
    - $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$
  - Multiplicative Identity: $x \otimes \bar{1} = x$

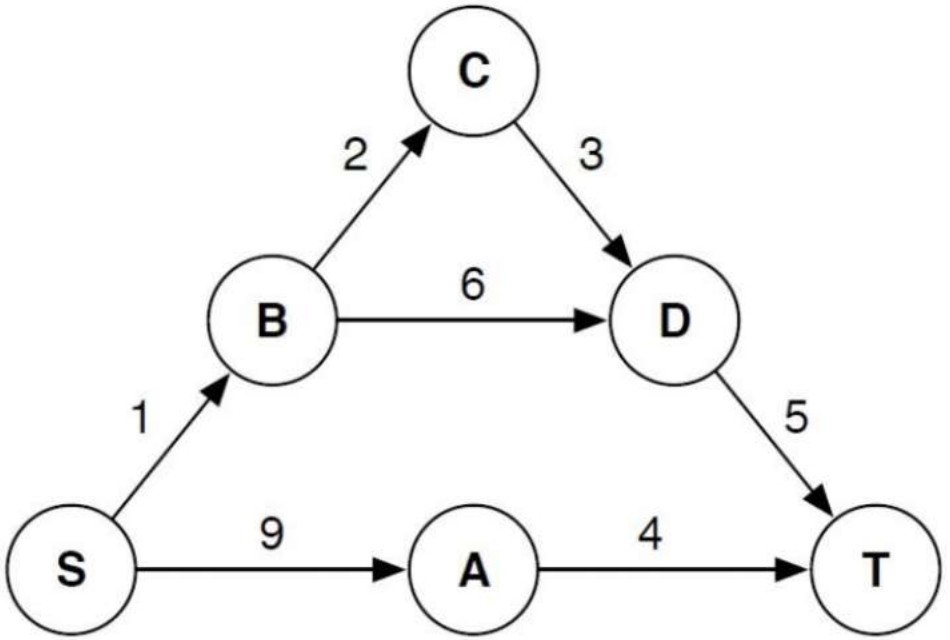
Why study Semirings?

In most areas that uses mathematics, understanding how a *field* works is more than enough.



However, when it comes to manipulating WFSTs, a knowledge of semirings is required. Also useful for **Generic problem solving**

Shortest Path from S to T



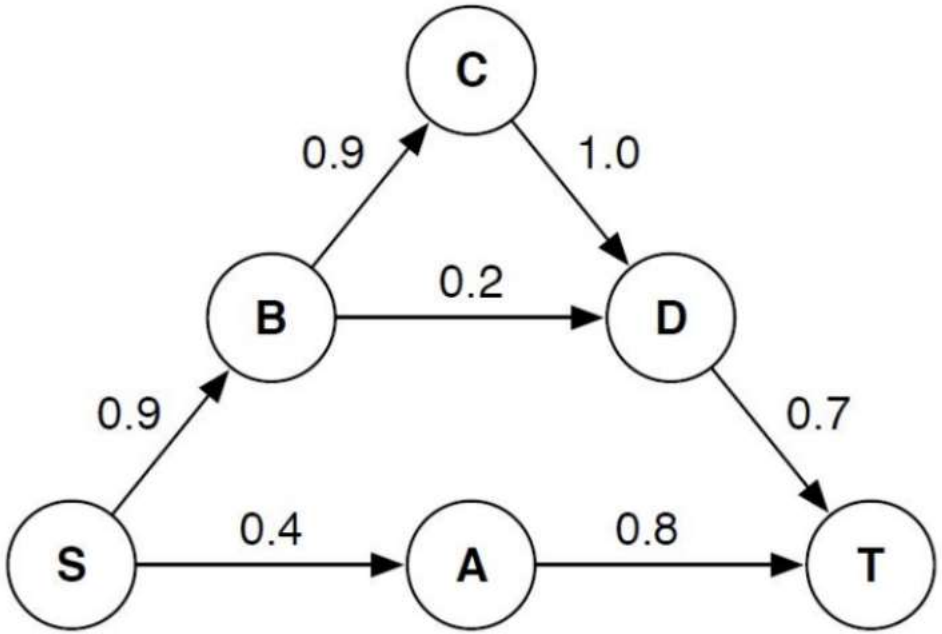
Step 1, Compute:

- $9 + 4 = 13$
- $1 + 6 + 5 = 12$
- $1 + 2 + 3 + 5 = 11$

Step 2, Then:

- $min\{13, 12, 11\} = 11$

Maximum Reliability from S to T



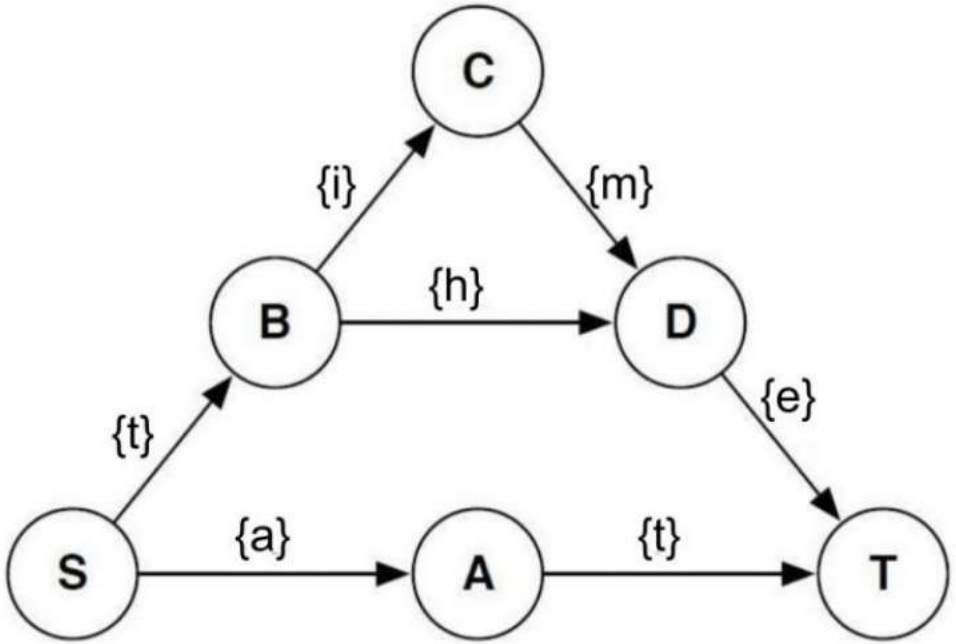
Step 1, Compute:

- $0.4 \times 0.8 = 0.32$
- $0.9 \times 0.2 \times 0.7 = 0.126$
- $0.9 \times 0.9 \times 1.0 \times 0.7 = 0.567$

Step 2, Then:

- $max\{0.32, 0.126, 0.567\} = 0.567$

Language leading from S to T in the Automaton



Step 1, Compute:

- $\{a\} \cdot \{t\} = \{at\}$
- $\{t\} \cdot \{h\} \cdot \{e\} = \{the\}$
- $\{t\} \cdot \{i\} \cdot \{m\} \cdot \{e\} = \{time\}$

Step 2, Then:

- $\bigcup\{\{at\}, \{the\}, \{time\}\} = \{at, the, time\}$

Generic Problem Solving

These problems were vastly different, but at the core, they are actually very much the same problem. Consider this:

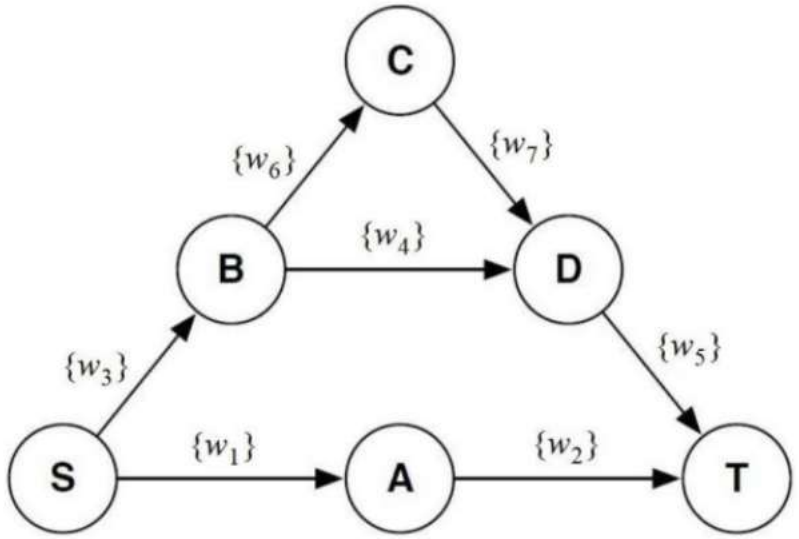
- $min\{(9+4),(1+6+5),(1+2+3+5)\}$
- $max\{(0.4\times 0.8),(0.9\times 0.2\times 0.7),(0.9\times 0.9\times 1.0\times 0.7)\}$
- $\bigcup\{\{a\}\cdot\{t\},\{t\}\cdot\{h\}\cdot\{e\},\{t\}\cdot\{i\}\cdot\{m\}\cdot\{e\}\}$

Is there a more general way of looking at the expressions above?

Generic Problem Solving

If we were to replace the inner operations with $\otimes$ and the outer operation with $\oplus$

- $\bigoplus \{\underbrace{\{w_1 \otimes w_2\}}_{w(\pi_1)}, \underbrace{\{w_3 \otimes w_4 \otimes w_5\}}_{w(\pi_2)}, \underbrace{\{w_3 \otimes w_6 \otimes w_7 \otimes w_5\}}_{w(\pi_3)}\}$



Even more compactly: $\bigoplus_{\pi \in P(S,T)} w(\pi)$

Common Examples of Semirings

- Probability: $([0,1], +, \times, 0, 1)$
- Strings: $(\Sigma^* \cup s_\infty, \wedge, \cdot, s_\infty, \varepsilon)$, where
  - Σ^*: A finite set of strings
  - s_∞: Infinite String
  - $\wedge$: Lowest common prefix $\quad [\{abcde\} \wedge \{bcf\} = \{bc\}]$
  - $\cdot$: Concatenation of 2 strings
- Tropical: $(\mathbb{R}, min, +, \infty, 0)$, where
  - min: The minimum function
- Log: $(\mathbb{R}, \oplus_{LOG}, +, \infty, 0)$, where
  - $x \oplus_{LOG} y = -log(e^{-x} + e^{-y})$

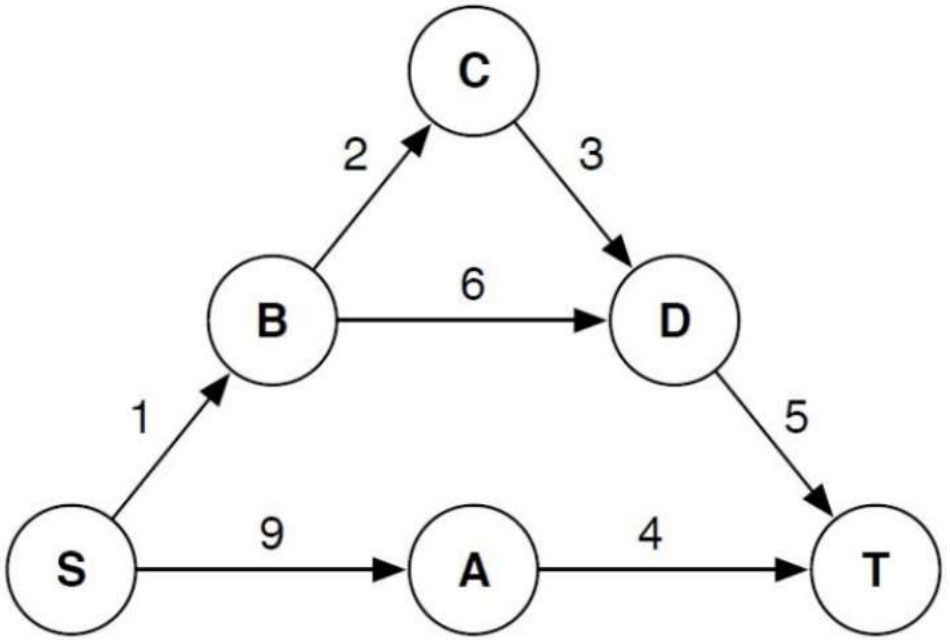
Semiring (*Definition*):

5 Tuple written as $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Where

- $\mathbb{K}$: A set containing some mathematical objects
- $\oplus$: Addition operation
- $\otimes$: Multiplication operation
- $\bar{0}$: Additive Identity. *Note*: $\bar{0} \in \mathbb{K}$
- $\bar{1}$: Multiplicative Identity. *Note*: $\bar{1} \in \mathbb{K}$

Shortest Path from S to T



Step 1, Compute:

- $9 + 4 = 13$
- $1 + 6 + 5 = 12$
- $1 + 2 + 3 + 5 = 11$

Step 2, Then:

- $min\{13, 12, 11\} = 11$

Proof of Tropical Semiring

Tropical Semiring (*Definition*):

5 Tuple written as $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- Addition:
  - Associativity:

$$x \oplus (y \oplus z) = min(x, min(y, z)) = min(min(x, y), z) = (x \oplus y) \oplus z$$

  - Commutativity:

$$x \oplus y = min(x, y) = min(y, x) = y \oplus x$$

  - Additive Identity:

$$x \oplus \bar{0} = min(x, \infty) = x$$

Proof of Tropical Semiring

Tropical Semiring (*Definition*):

5 Tuple written as $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- Multiplication:
  - Associativity:

$$x \otimes (y \otimes z) = x + (y + z) = (x + y) + z = (x \otimes y) \otimes z$$

  - Multiplicative Identity:

$$x \otimes \bar{1} = x + 0 = x$$

Proof of Tropical Semiring

Tropical Semiring (*Definition*):

5 Tuple written as $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Left Distributivity of Multiplication, WLOG suppose $x \leq y$:

- $z \otimes (x \oplus y) = z + min(x, y) = z + x$
- $(z \otimes x) \oplus (z \otimes y) = min(z + x, z + y) = z + x = z \otimes (x \oplus y)$

Right Distributivity of Multiplication

- $(x \oplus y) \otimes z = min(x, y) + z = x + z$
- $(x \otimes z) \oplus (y \otimes z) = min(x + z, y + z) = x + z = (x \oplus y) \otimes z$

Proof of Log Semiring

Log Semiring (*Definition*):

5 Tuple written as $(\mathbb{R}, \oplus_{LOG}, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Associativity of Addition ($x \oplus_{LOG} y = -log(e^{-x} + e^{-y})$)

$$
\begin{aligned}
x \oplus_{LOG} (y \oplus_{LOG} z) &= -log(e^{-x} + e^{-(y \oplus_{LOG} z)}) \\
&= -log(e^{-x} + e^{-(-log(e^{-y}+e^{-z}))}) \\
&= -log(e^{-x} + e^{log(e^{-y}+e^{-z})}) \\
&= -log(e^{-x} + e^{-y} + e^{-z}) \\
&= -log(e^{log(e^{-x}+e^{-y})} + e^{-z}) \\
&= -log(e^{-(x \oplus_{LOG} y)} + e^{-z}) \\
&= (x \oplus_{LOG} y) \oplus_{LOG} z
\end{aligned}
$$

Proof of Log Semiring

Log Semiring (*Definition*):

5 Tuple written as $(\mathbb{R}, \oplus_{LOG}, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- Addition
  - Commutativity:

    $$x \oplus_{LOG} y = -log(e^{-x} + e^{-y}) = -log(e^{-y} + e^{-x}) = y \oplus_{LOG} x$$

  - Additive Identity:
    $x \oplus_{LOG} \bar{0} = -log(e^{-x} + e^{-\infty}) = -log(e^{-x}) = x$
- Multiplication:
  - Associativity:
    $x \otimes (y \otimes z) = x + (y + z) = (x + y) + z = (x \otimes y) \otimes z$
  - Additive Identity: $x \otimes \bar{1} = x + 0 = x$

Basic Operations

WFST over semiring $S = (\mathbb{K}, \oplus, \otimes, \bar{1}, \bar{0})$ (Definition):

$$T(x, y) = \bigoplus_{\pi \in P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi])$$

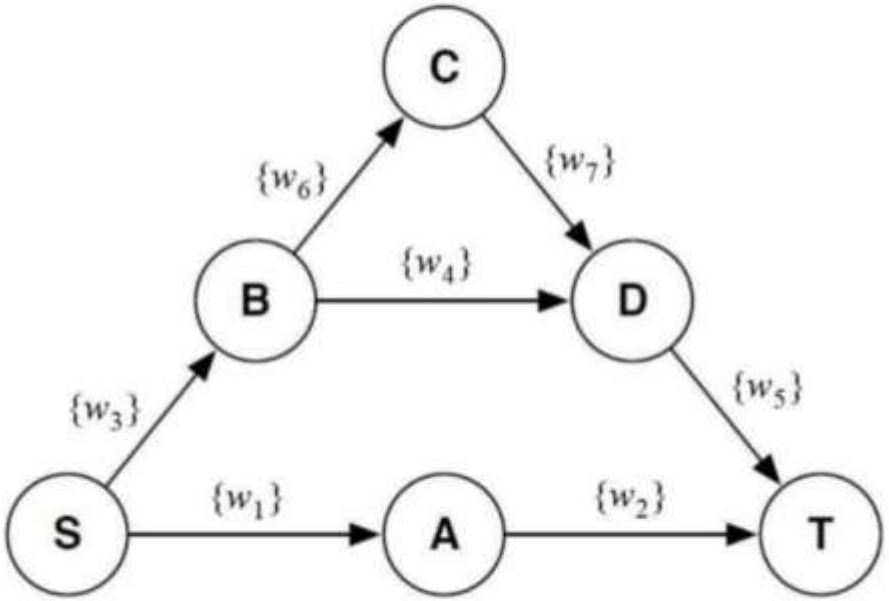


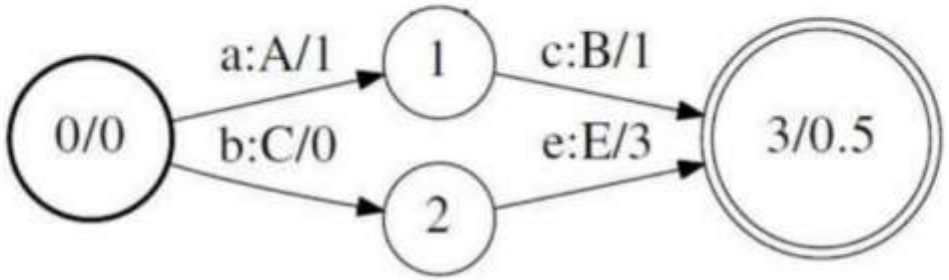
Figure: WFSA

Recall that: $\bigoplus_{\pi \in P(S,T)} w(\pi)$

The expression above is analogous to the definition of WFST over a Semiring.

The *extra* $\lambda(p[\pi])$ and $\rho(n[\pi])$ are the weights of the initial and final states respectively.

Given a WFSTs over a **Tropical** Semiring: $(\mathbb{R}, min, +, \infty, 0)$



Consider $\pi_0 = (0, a, A, 1, 1), (1, c, B, 1, 3),\ x = \{ac\},\ y = \{AB\}$

$$
\begin{aligned}
T(x,y) &= \bigoplus_{\pi \in P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi]) \\
&= \lambda(p[\pi_0]) \otimes w[\pi_0] \otimes \rho(n[\pi_0]) \\
&= \lambda(0) \otimes (1 \otimes 1) \otimes \rho(3) \\
&= 0 + 1 + 1 + 0.5 = 2.5
\end{aligned}
$$

Basic Operations

WFST over semiring $S = (\mathbb{K}, \oplus, \otimes)$ (Definition):

$$T(x, y) = \bigoplus_{\pi \in P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi])$$

Kleene closure:

- Modifies the automaton so that the set of symbol sequences or transductions is sequentially repeated zero or more times

Union:

- Combines two automata in **parallel**

Concatenation:

- Combines two automata in **series**

Basic Operations

WFST over semiring $S = (\mathbb{K}, \oplus, \otimes)$ (Definition):

$$T(x, y) = \bigoplus_{\pi \in P(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \otimes \rho(n[\pi])$$

Inverse:

- Exchanging *input* and *output* symbols

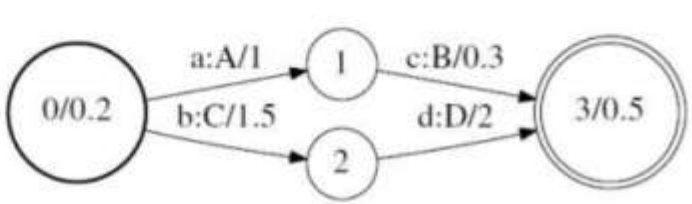
Reversal:

- Assigns to each pair of strings (x, y) what T assigns to its mirror images $(\tilde{x}, \tilde{y})$

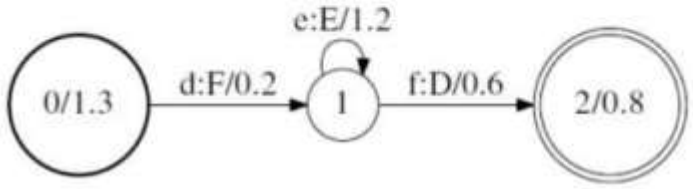
Projection:

- Converting a transducer into an acceptor by omitting input or output labels

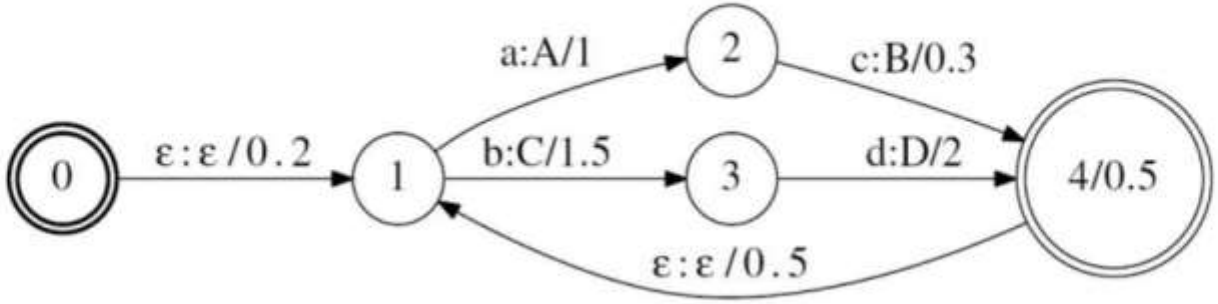
An example of a Kleene Closure



(a) Example WFST T_A.



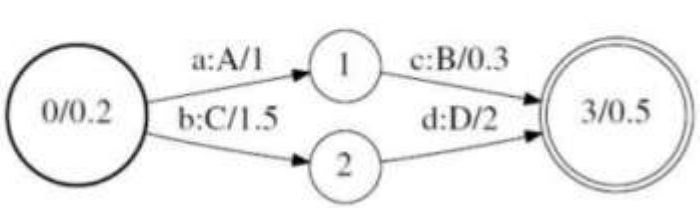
(b) Example WFST T_B.



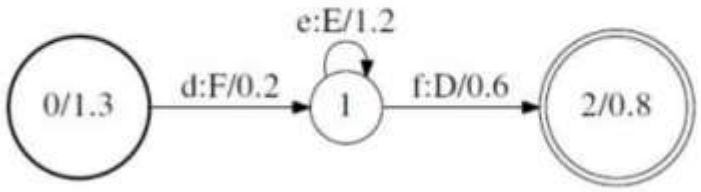
(c) Kleene Closure: T_A^*.

Definition: $T^*(x, y) = \bigoplus_{n=0}^{\infty} T^n(x, y)$

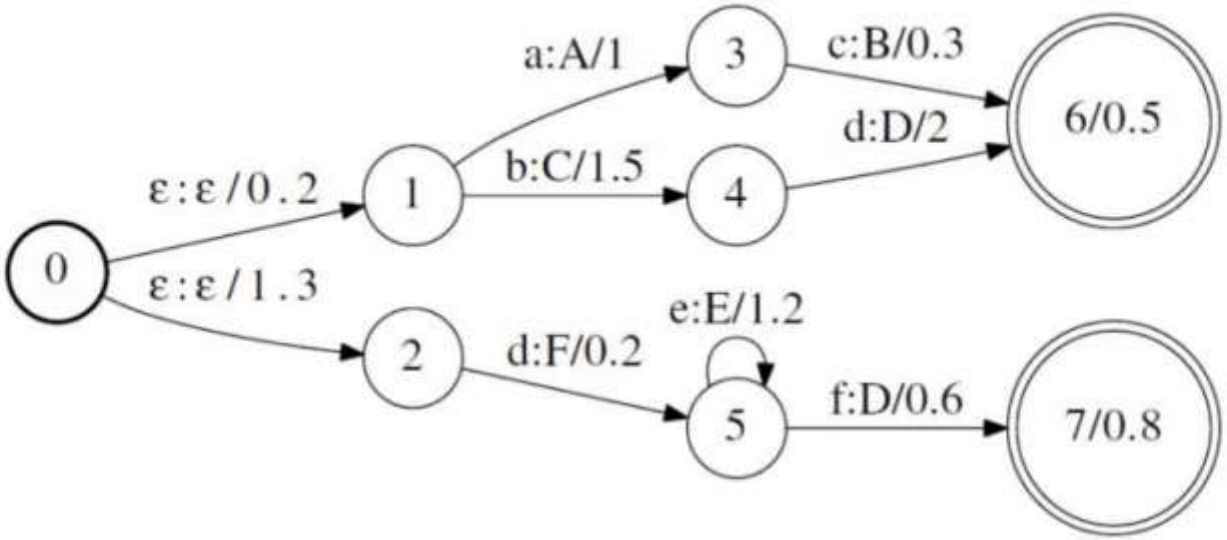
An example of a Union



(a) Example WFST T_A.



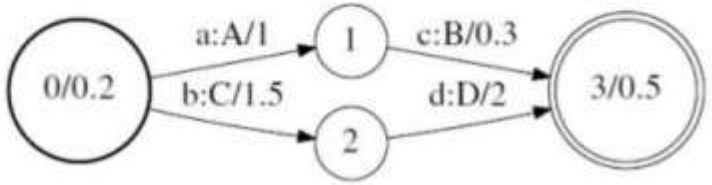
(b) Example WFST T_B.



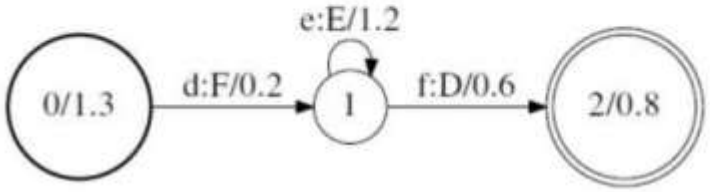
(d) Union: $T_A + T_B$.

Definition: $(T_A \oplus T_B)(x, y) = T_A(x, y) \oplus T_B(x, y)$

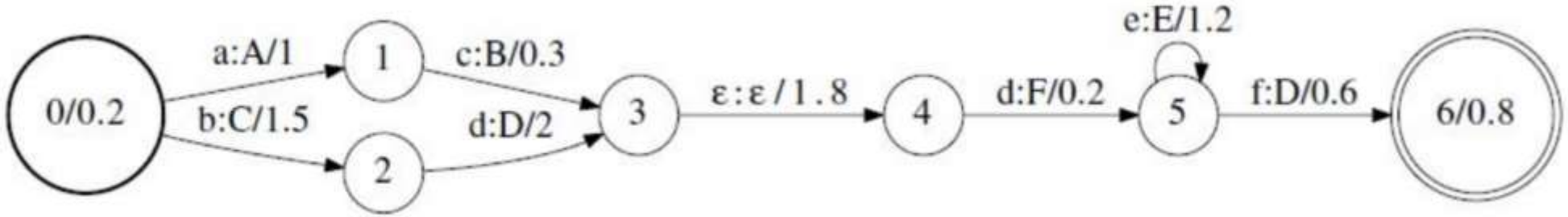
An example of a Concatenation



(a) Example WFST T_A.



(b) Example WFST T_B.

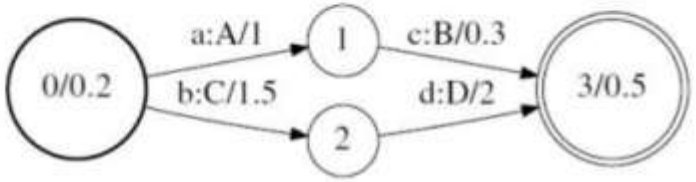


(e) Concatenation: $T_A \times T_B$.

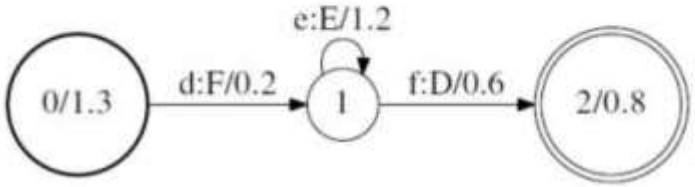
Definition:

$$(T_A \otimes T_B)(x, y) = \bigoplus_{x=x_A, x_B; y=y_A, y_B} T_A(x_A, y_A) \otimes T_B(x_B, y_B)$$

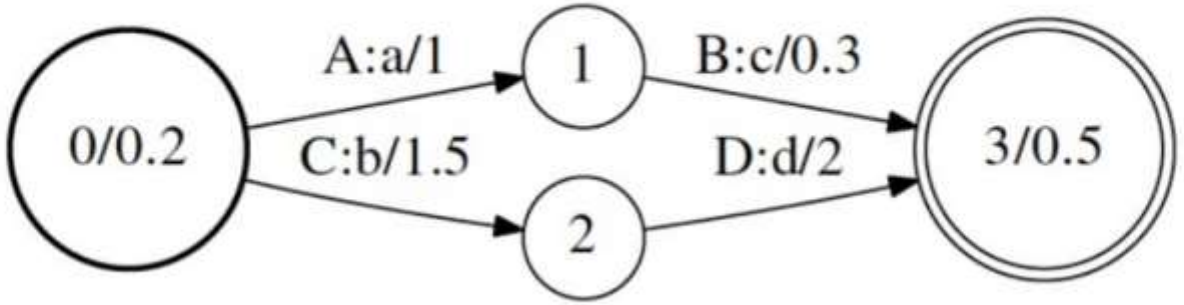
An example of an Inverse



(a) Example WFST T_A.



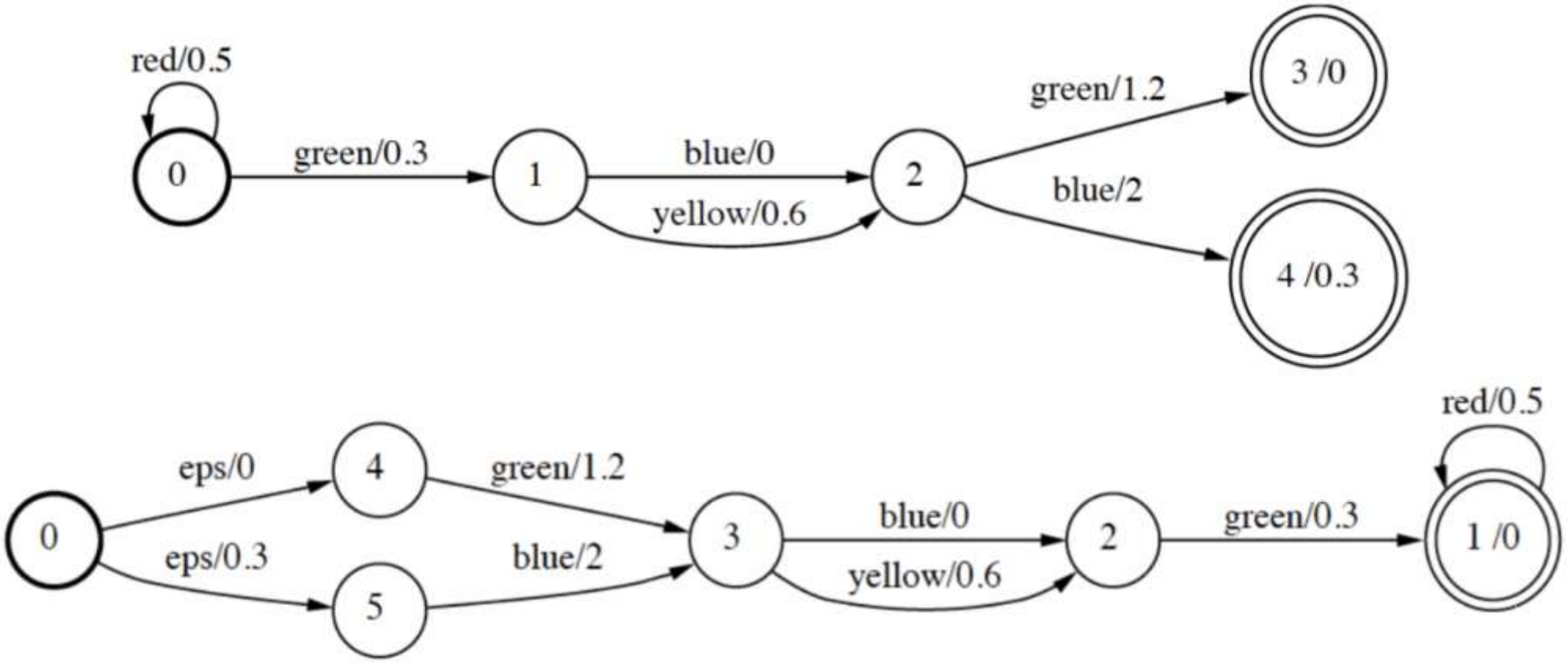
(b) Example WFST T_B.



(b) Inversion of T_A.

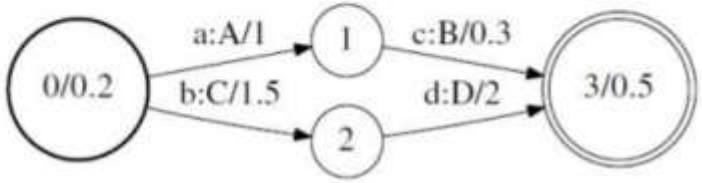
Definition: $T^{-1}(x, y) = T(y, x)$

An example of a Reversal

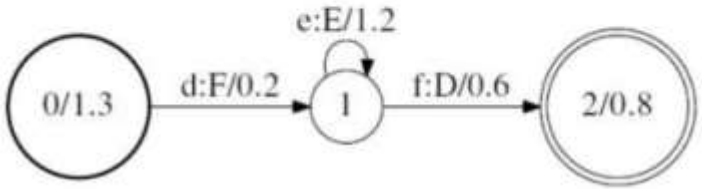


Definition: $\tilde{T}(x, y) = T(\tilde{y}, \tilde{x})$

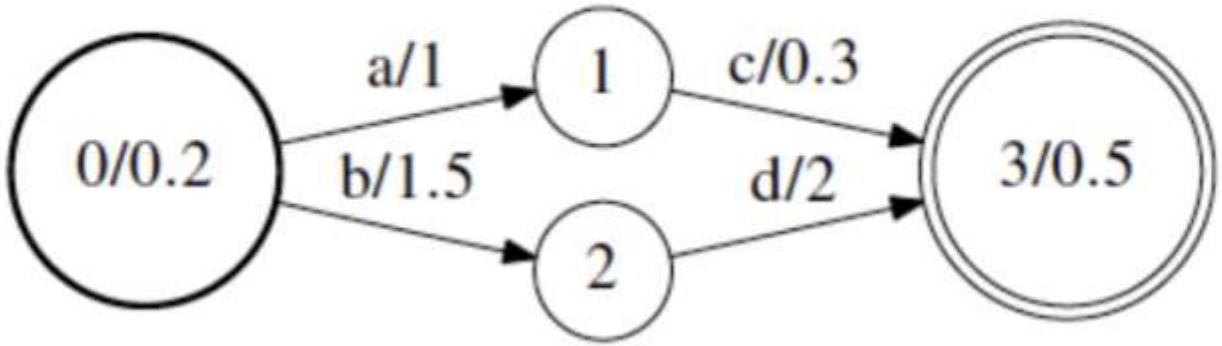
An example of a Projection



(a) Example WFST T_A.



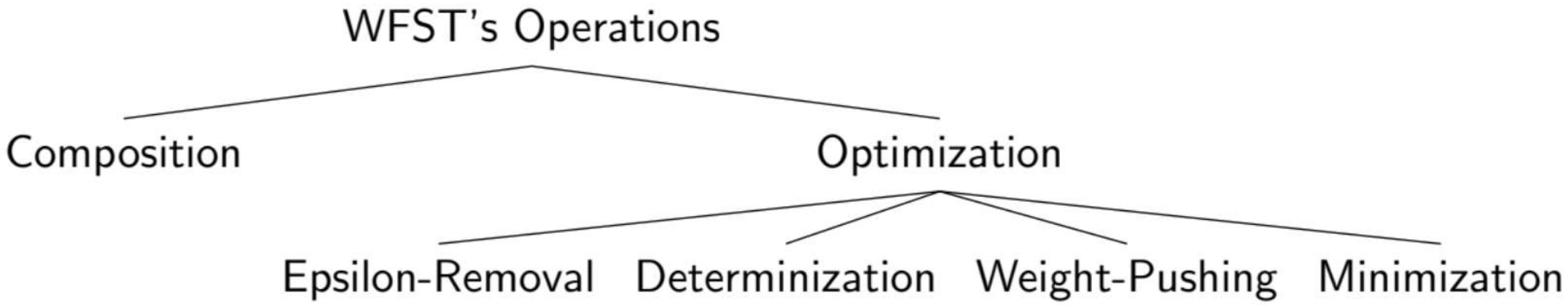
(b) Example WFST T_B.



(a) Projection of T_A.

Definition: $\downarrow T(x) = \bigoplus_y T(y, x)$

2 main Operations of WFSTs

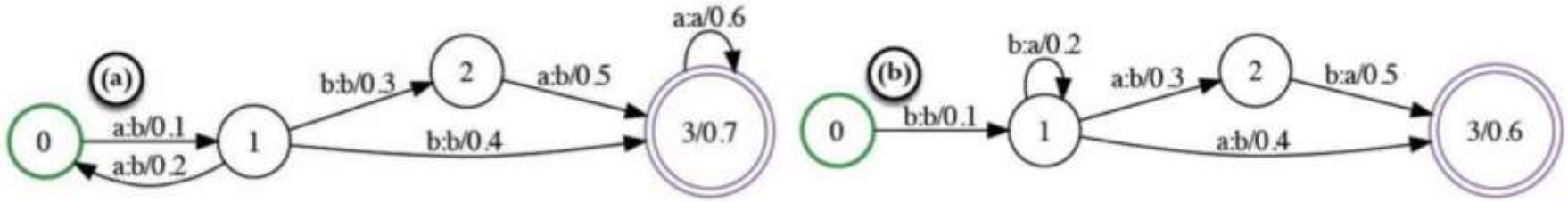


Rationale for Composition, an example

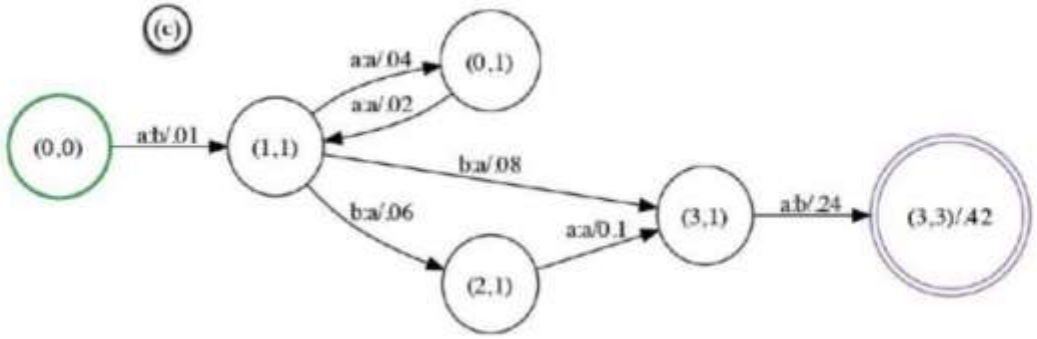
- T_A converts a **sequence of letters** so that they are all **uppercase letters**
- T_B converts the sequence of **uppercase letters** to a **sequence of specific words** that matches the letters
- $T = T_A \circ T_B$ converts a **sequence of letters** into a **sequence of specific words**
- The resulting transducer may be harder to construct from scratch
- So given a complicated transducer, it can be broken down into many simple transducers
- Composition simplifies construction of complicated transducers

Composition

- Combining two related transducers into one single transducer that represents "*a set of transductions cascaded with the original two transducers*" (*Takaaki Hori,2013*). Given 2 WFST over the *Probability Semiring*:



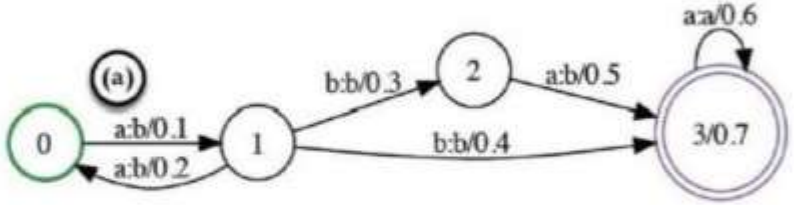
- We obtain:



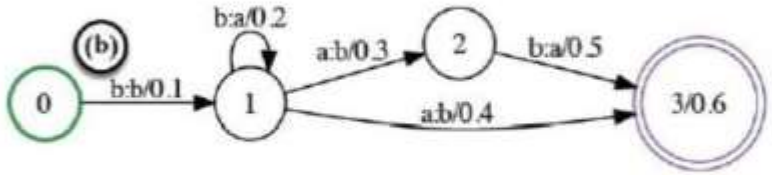
Given 2 transducers:

$$T_A : (\Sigma, \Delta, Q_1, E_1, I_1, F_1, \lambda_1, \rho_1), T_B : (\Delta, \Omega, Q_2, E_2, I_2, F_2, \lambda_2, \rho_2)$$

$$(T_A \circ T_B)(x, y) = \bigoplus_{z \in \Delta} T_A(x, z) \otimes T_B(z, y)$$

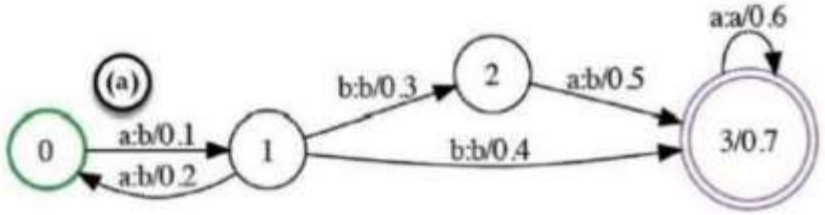


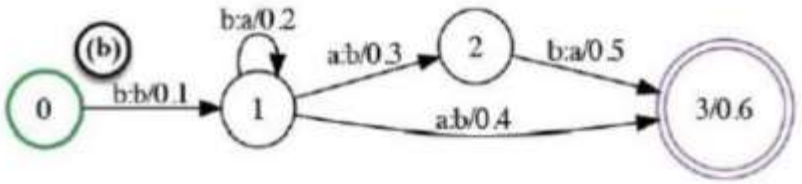
I	Input/Output/Weight
$(0_A, 1_A)$	$a : b / 0.1$
$(1_A, 0_A)$	$a : b / 0.2$
$(1_A, 2_A)$	$b : b / 0.3$
$(1_A, 3_A)$	$b : b / 0.4$
$(2_A, 3_A)$	$a : b / 0.5$
$(3_A, 3_A)$	$a : a / 0.6$



I	Input/Output/Weight
$(0_B, 1_B)$	$b : b / 0.1$
$(1_B, 1_B)$	$b : a / 0.2$
$(1_B, 2_B)$	$a : b / 0.3$
$(1_B, 3_B)$	$a : b / 0.4$
$(2_B, 3_B)$	$b : a / 0.5$

Composition I





- First iteration, consider $(0_A, 0_B)$

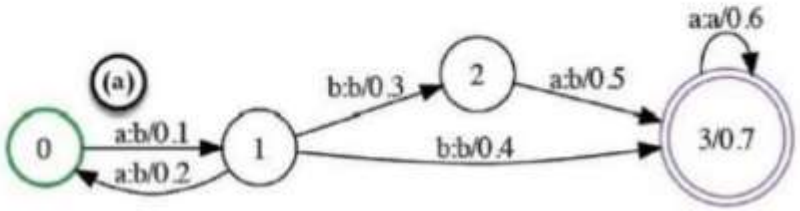
Input/Output/Weight of A	Input/Output/Weight of B
$a : b \,/\, 0.1 \rightarrow 1_A$	$b : b \,/\, 0.1 \rightarrow 1_B$

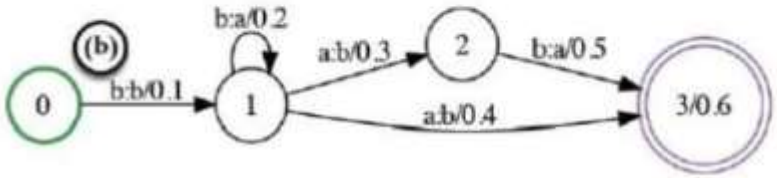
After Composition
$a : b \,/\, 0.01 \rightarrow (1_A, 1_B)$

Queue K
$(0_A, 0_B)$

Queue $K+1$
$(1_A, 1_B)$

Composition II





- Second iteration, consider $(1_A, 1_B)$

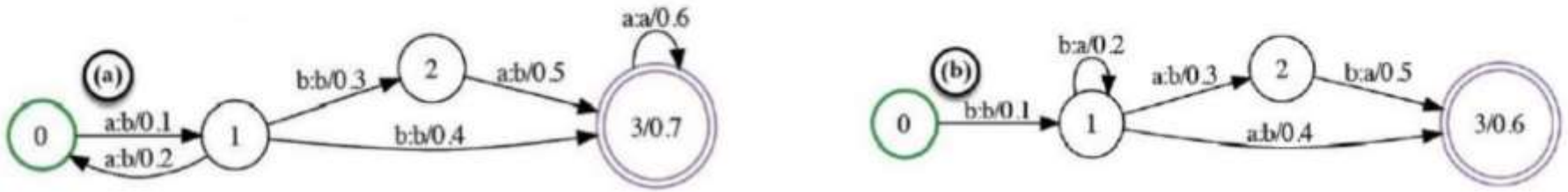
$a:b/0.2 \rightarrow 0_A$	$b:a/0.2 \rightarrow 1_B$
$b:b/0.3 \rightarrow 2_A$	~~$a:b/0.3 \rightarrow 2_B$~~
$b:b/0.4 \rightarrow 3_A$	~~$a:b/0.4 \rightarrow 3_B$~~

After Composition
$a:a/0.04 \rightarrow (0_A, 1_B)$
$b:a/0.06 \rightarrow (2_A, 1_B)$
$b:a/0.08 \rightarrow (3_A, 1_B)$

Queue K
$(1_A, 1_B)$

Queue $K+1$
$(0_A, 1_B)$
$(2_A, 1_B)$
$(3_A, 1_B)$

Composition III



- Third iteration, consider $(0_A, 1_B)$

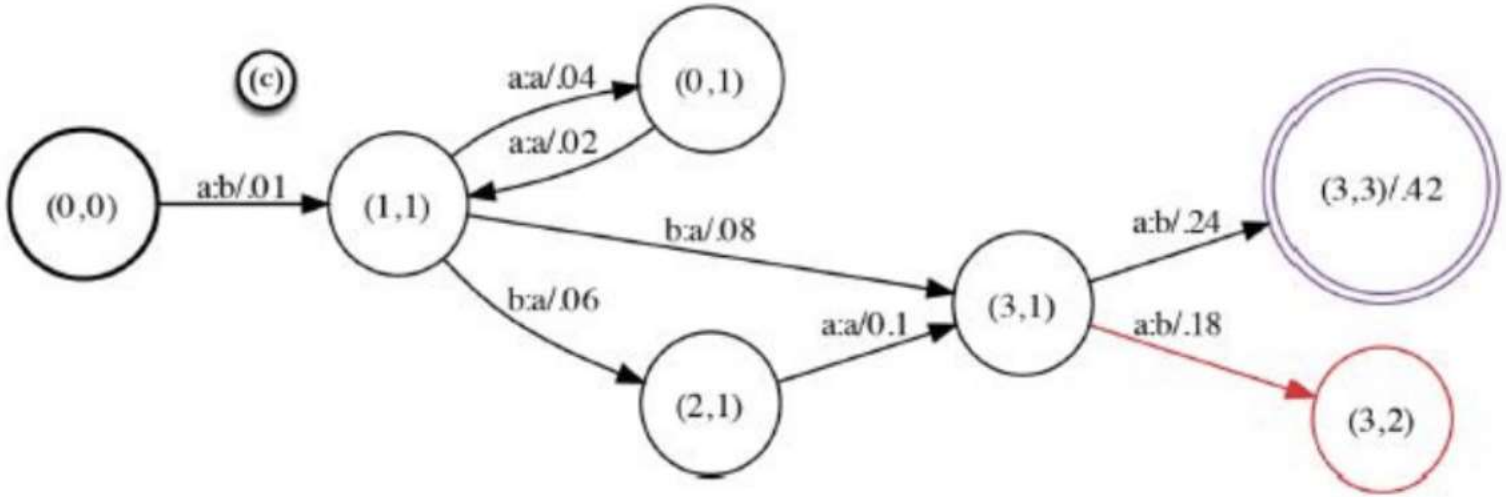
$a : b \,/\, 0.1 \rightarrow 1_A$	$b : a \,/\, 0.2 \rightarrow 1_B$
	~~$a : b \,/\, 0.3 \rightarrow 2_B$~~
	~~$a : b \,/\, 0.4 \rightarrow 3_B$~~

After Composition
$a : a \,/\, 0.02 \rightarrow (1_A, 1_B)$

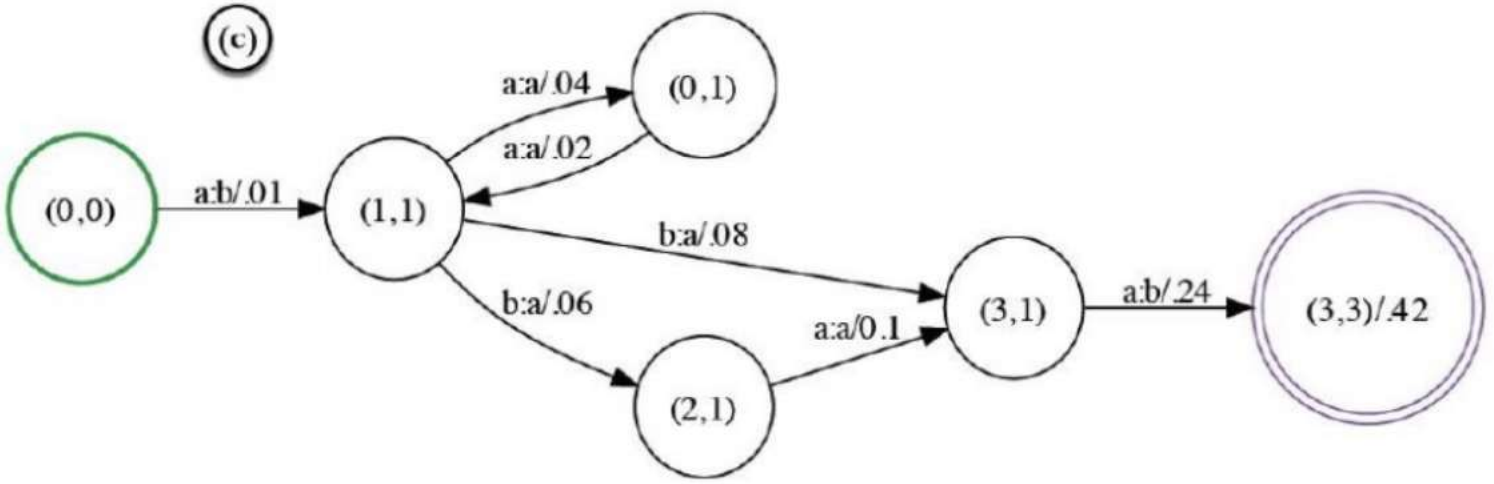
Queue K
$(0_A, 1_B)$

Queue $K+1$
$(2_A, 1_B)$
$(3_A, 1_B)$

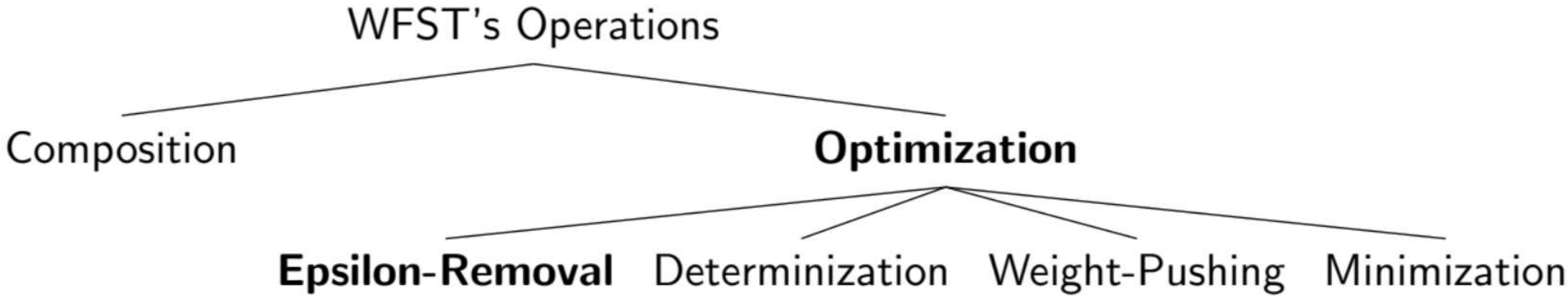
We obtain the new transducer, $T = T_A \circ T_B$ with an extra Dead node $(3, 2)$



Trimming away the node, we arrive at our previous result:



$(T_A \circ T_B)(x, y) : (\Sigma, \Omega, Q, E, I, F, \lambda, \rho)$ Where $Q \subseteq Q_1 \times Q_2$


WFST's Operations
Composition
Optimization
Epsilon-Removal
Determinization
Weight-Pushing
Minimization

Introduction to Epsilon-Removal

Removes ε-transitions from the input transducer and produces a new, epsilon-free transducer equivalent to the original

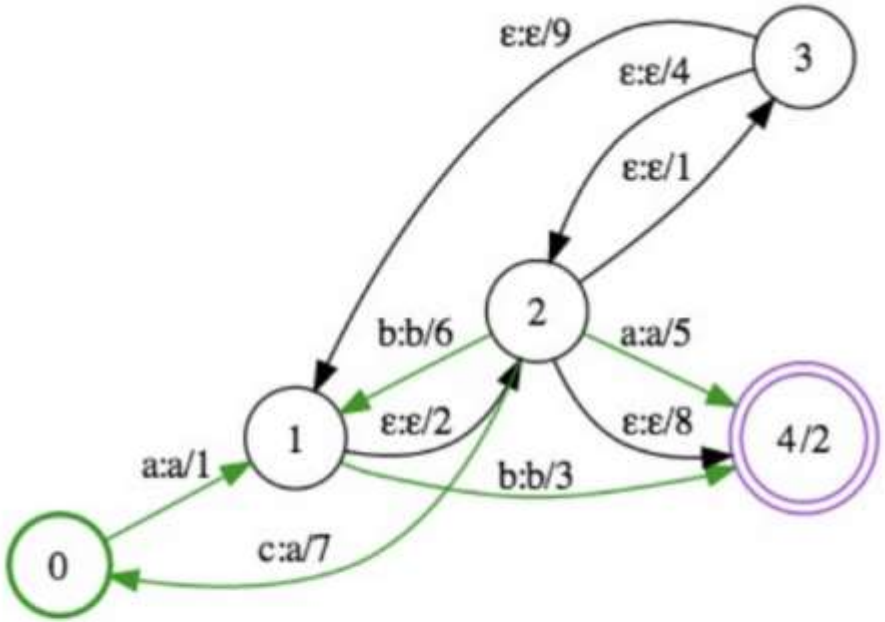


Figure: Before ε-Removal

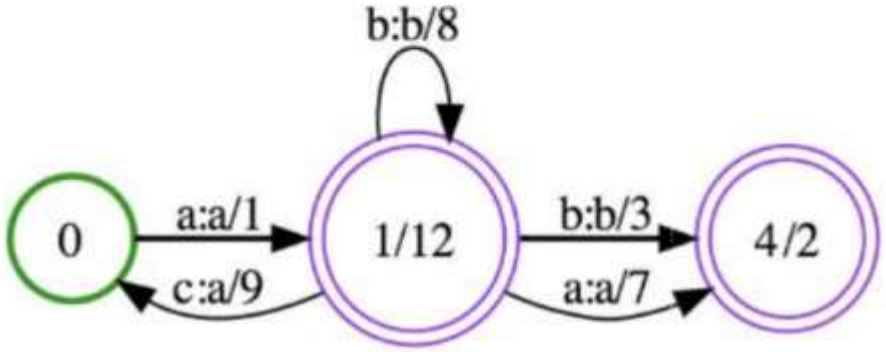


Figure: After ε-Removal

Intuition of Algorithm

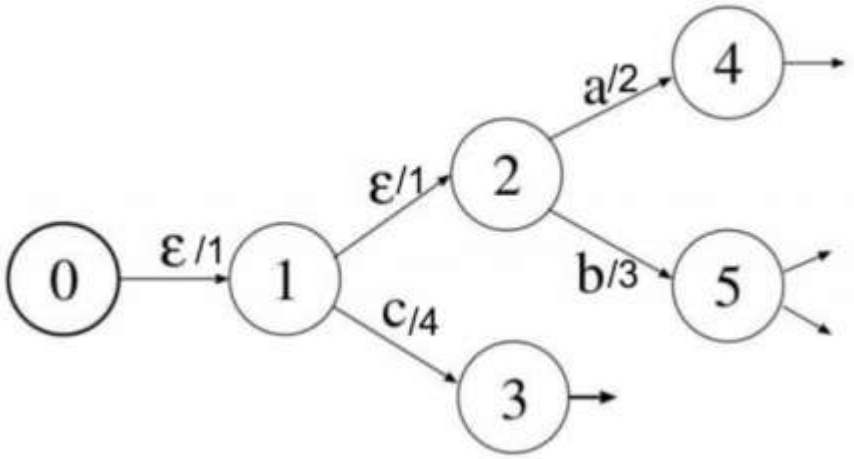


Figure: Part of Original FA

Consider the following *paths*:

1. $0 \rightarrow 1 \rightarrow 2 \rightarrow 4$
   - Requires input {a} only
2. $0 \rightarrow 1 \rightarrow 2 \rightarrow 5$
   - Requires input {b} only
3. $0 \rightarrow 1 \rightarrow 3$
   - Requires input {c} only

Intuition of Algorithm

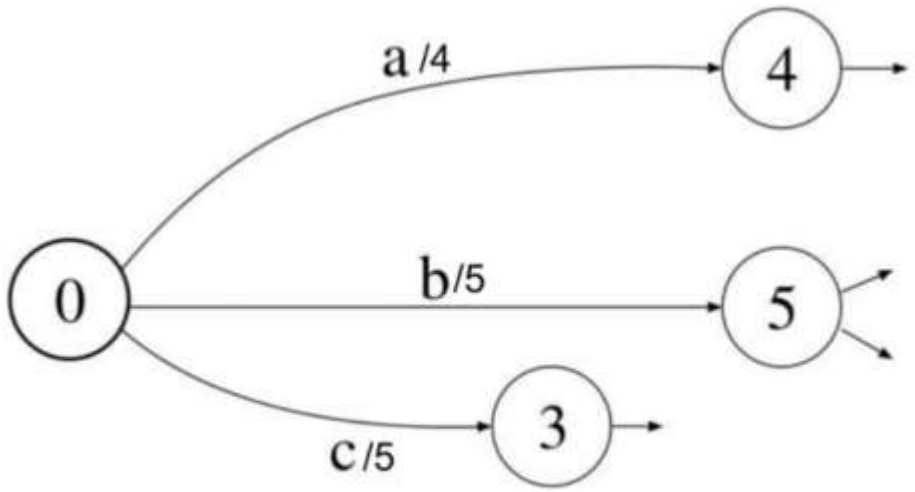


Figure: "ε-Closure" of FA

Remove redundant ε-transitions:

1. $0 \rightarrow 1 \rightarrow 2 \rightarrow 4$
   - $0 \rightarrow 4$
2. $0 \rightarrow 1 \rightarrow 2 \rightarrow 5$
   - $0 \rightarrow 5$
3. $0 \rightarrow 1 \rightarrow 3$
   - $0 \rightarrow 3$

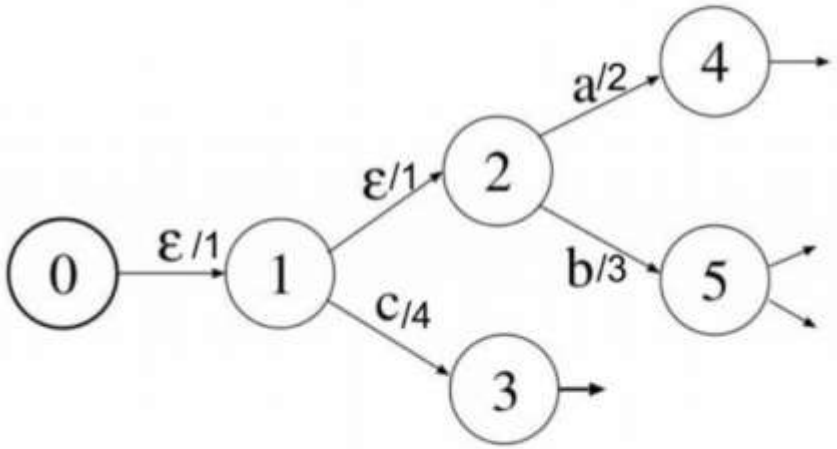
This step is to calculate the ε-*closure* of each state.

Epsilon-Removal Worked Example I

Weighted ε-closure (*Definition*)

$$C(p) = \{(q, w) : P(p, \varepsilon, q) \neq \emptyset, w = \delta_\varepsilon[p, q]\} \quad \forall p \in Q$$

Where $\delta_\varepsilon[p, q] = \bigoplus_{\pi \in P(p,\varepsilon,q)} w(\pi)$



$C(0)$ contains $(1, w_1)$ and $(2, w_2)$

- $w_1 = \delta_\varepsilon[0, 1] = 1$
- $w_2 = \delta_\varepsilon[0, 2] = 1 \otimes 1 = 2$

Thus $C(0) = \{(1, 1), (2, 2)\}$

Figure: WFSA over a Tropical SR

Epsilon-Removal Worked Example II

Augmenting all transitions associated at state 0

1. $\{(p, a, b, \delta_\varepsilon[p, q] \otimes w, r) : (q, a, b, w, r) \in E, (a, b) \neq (\varepsilon, \varepsilon)\}$
2. If $\exists(q, w)$ s.t $q \in F : w(p) \leftarrow w(p) \oplus (\delta_\varepsilon[p, q] \otimes \rho(q))$

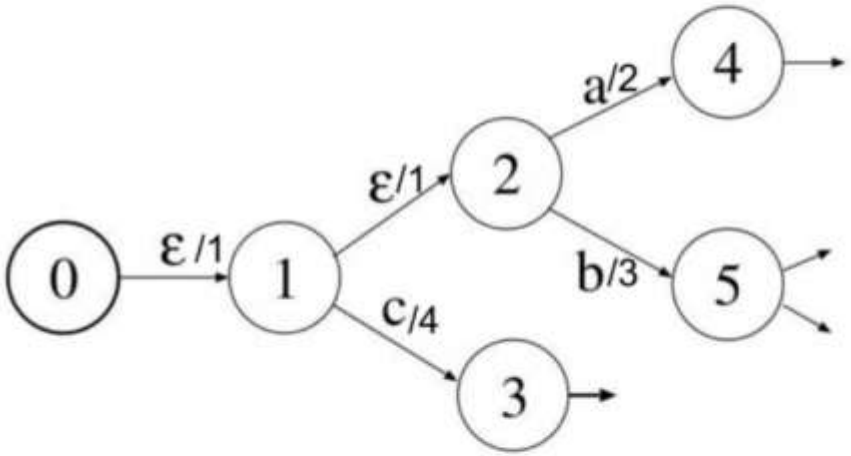


Figure: "ε-Closure" of FA

Consider all transitions that begin with "1" and "2", that **DO NOT** contain any ε-transitions

- $(2,a,2,4) \rightarrow (0,a,\delta_\varepsilon[0,2] \otimes 2,4)$
- $(2,b,3,5) \rightarrow (0,b,\delta_\varepsilon[0,2] \otimes 3,5)$
- $(1,c,4,3) \rightarrow (0,c,\delta_\varepsilon[0,1] \otimes 4,3)$

Epsilon-Removal Worked Example III

Augmenting all transitions associated at state 0

1. $\{(p, a, b, \delta_\varepsilon[p, q] \otimes w, r) : (q, a, b, w, r) \in E, (a, b) \neq (\varepsilon, \varepsilon)\}$
2. $\delta_\varepsilon[0, 2] = 2$ and $\delta_\varepsilon[0, 1] = 1$

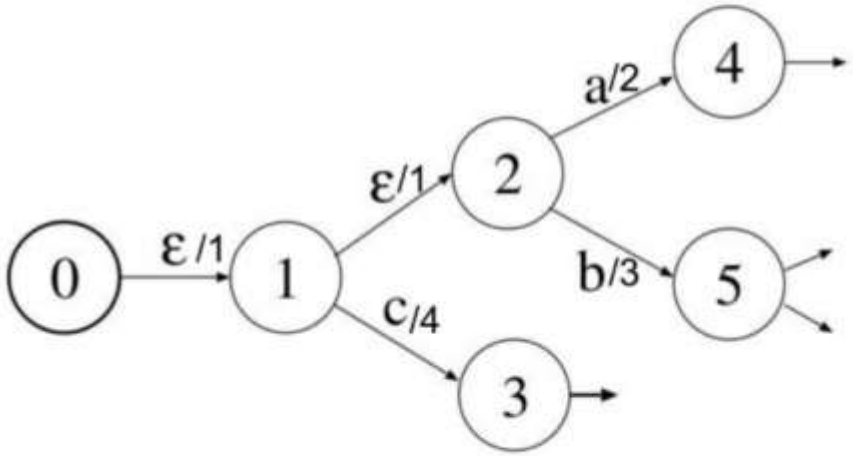


Figure: "ε-Closure" of FA

Replacing the 3 transitions with:

- (2,a,2,4) $\rightarrow$ (0,a,4,4)
- (2,b,3,5) $\rightarrow$ (0,b,5,5)
- (1,c,4,3) $\rightarrow$ (0,c,5,3)

Epsilon-Removal Worked Example IV

Removing the states 1 and 2 yields:

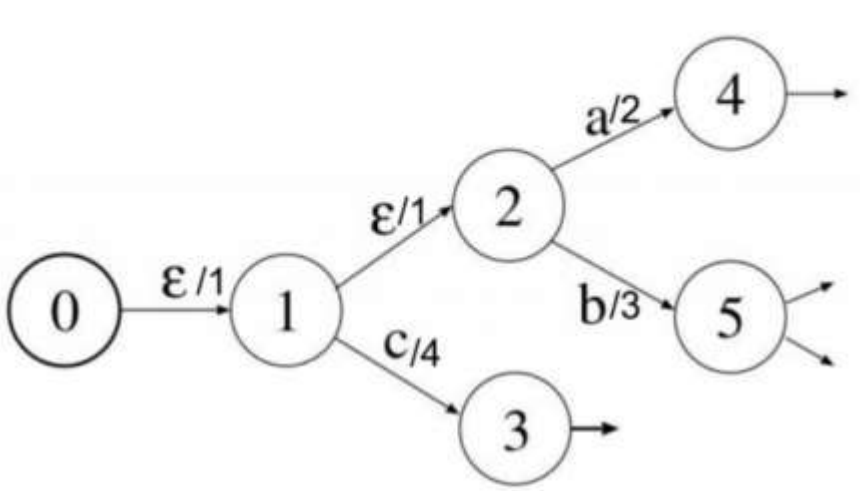


Figure: Before ε-Removal

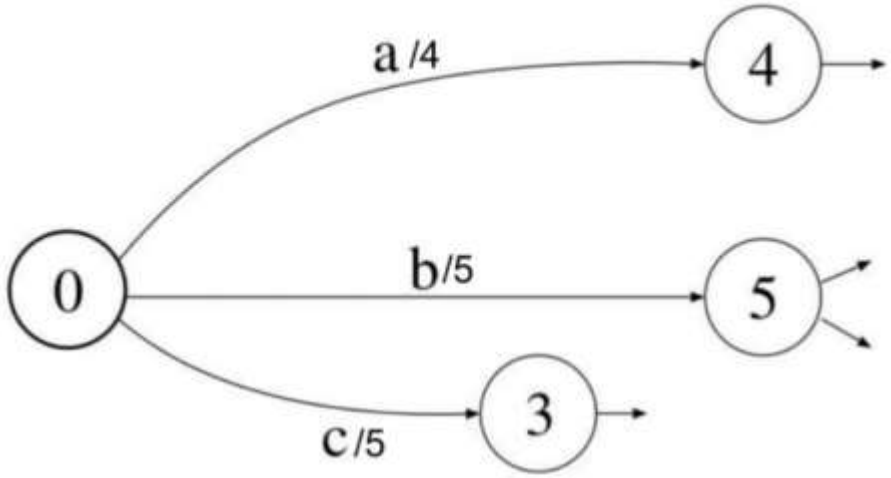
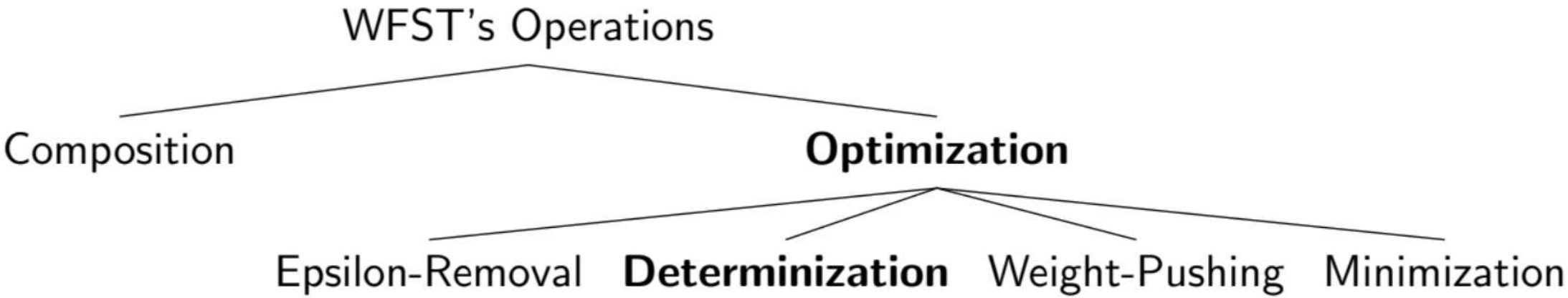


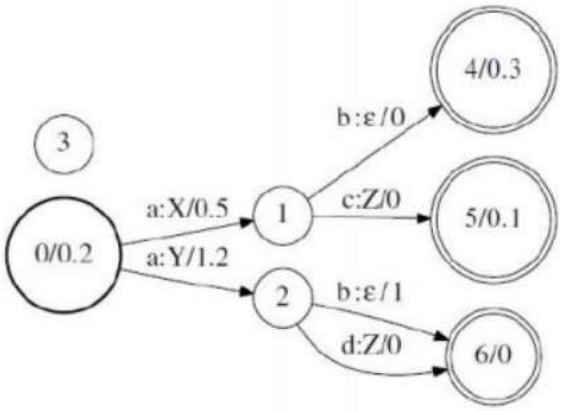
Figure: After ε-Removal

The new transitions: (0,*a*,4,4), (0,*b*,5,5) and (0,*a*,4,3)

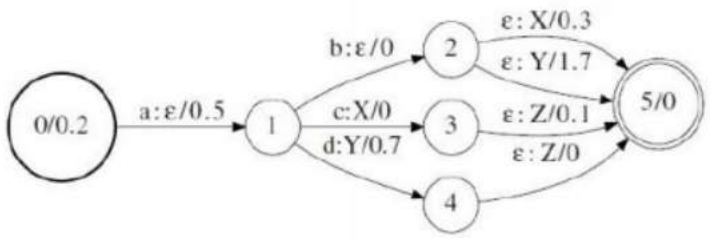

WFST's Operations
Composition
Optimization
Epsilon-Removal
Determinization
Weight-Pushing
Minimization

NFA (Informal): At certain states, cannot determine where it will go to, hence the destination states is a **powerset**, instead of a single state

For example: Given the input *ab* there are 2 possible states 4 or 6.

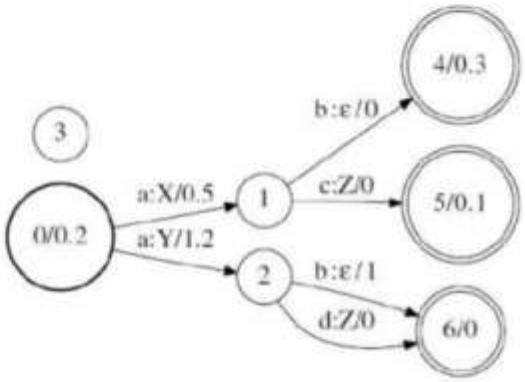


(a) Non-deterministic WFST: T.

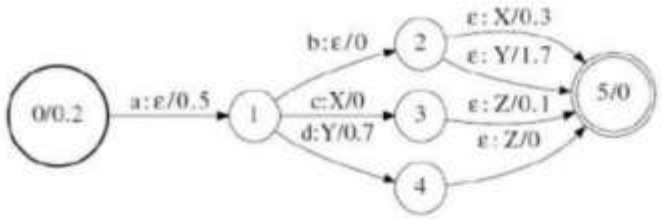


(c) Determinized WFST: det(T).

Description of Algorithm



(a) Non-deterministic WFST: T.



(c) Determinized WFST: det(T).

- States are grouped based on the transitions input symbols
- Thus, the resulting transducer have *UNIQUE* inputs and *UNIQUE* paths
- Multiple final states are compressed into a single, weighted final state
- final emission function in each final state is replaced with epsilon transitions

Determinization: Worked Example

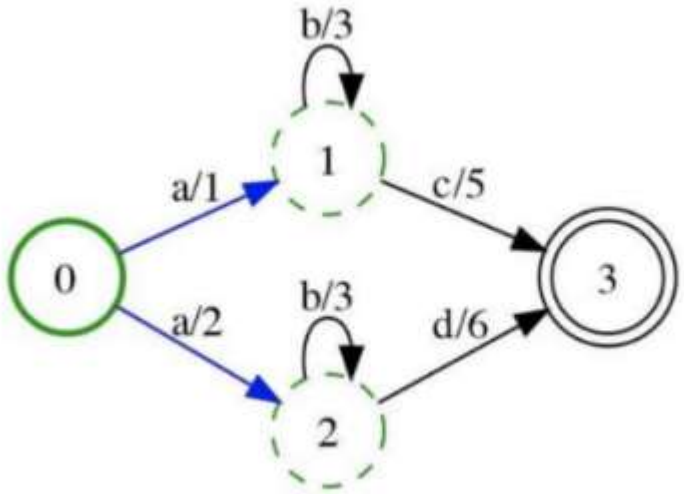


Figure: Original WFSA over Tropical Semiring

- First label initial state with weight $\bar{1}$.
- Then start by focusing on all the transitions with the same input.
- Compute total weight of given input "a":
  - $w(e_1) = \bar{1} \otimes 1 = 0 + 1 = 1$
  - $w(e_2) = \bar{1} \otimes 2 = 0 + 2 = 2$
  - $w(e_1) \oplus w(e_2) = min(1, 2) = 1$
- This is the new arc weight $w(e_1') = 1$

Determinization: Worked Example

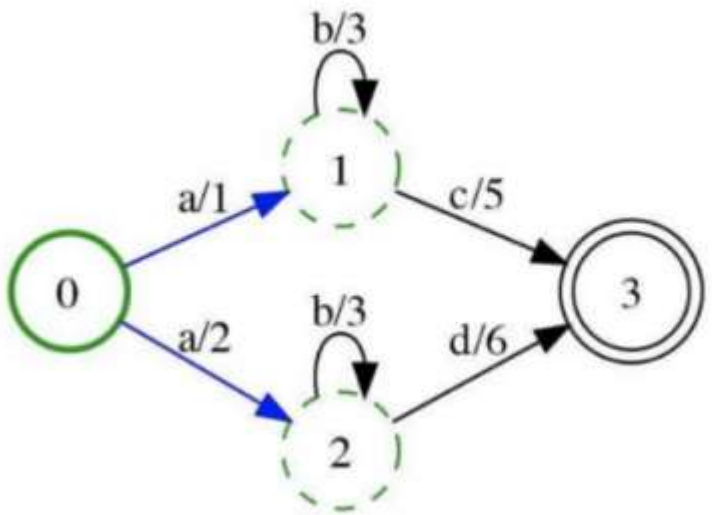


Figure: Original WFSA over Tropical Semiring

- Start by computing residual weights:
  - $w_r(e_i) = w(e_j')^{-1} \otimes w(p[e_i]) \otimes w(e_i)$
  - $w_r(e_1) = -1 + 0 + 1 = 0$
  - $w_r(e_2) = -1 + 0 + 2 = 1$
  - $n[e_1] =$"1" and $n[e_2] =$"2"
- Group using a new notation:
  - $((1, 0), (2, 1))$, the new state
- Then the new transition is:
  - $\{(0, 0), a, 1, ((1, 0), (2, 1))\}$

Determinization: Worked Example

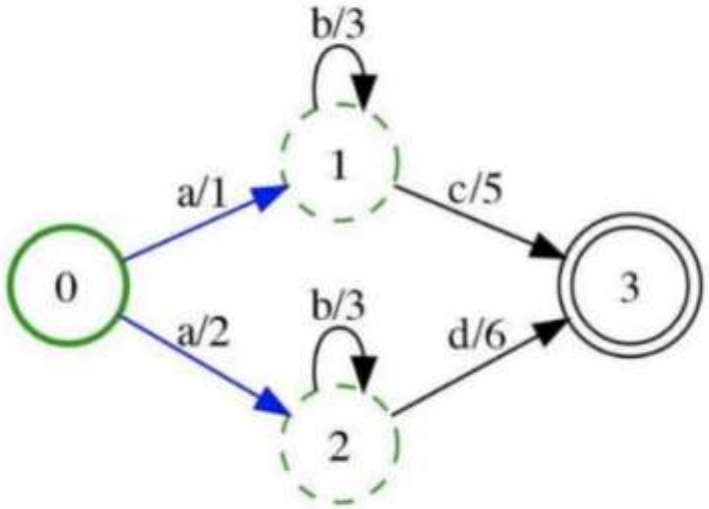


Figure: Original WFSA over Tropical Semiring

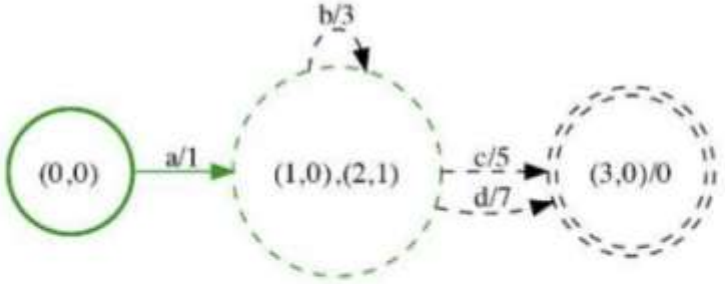


Figure: After first update

Determinization: Worked Example

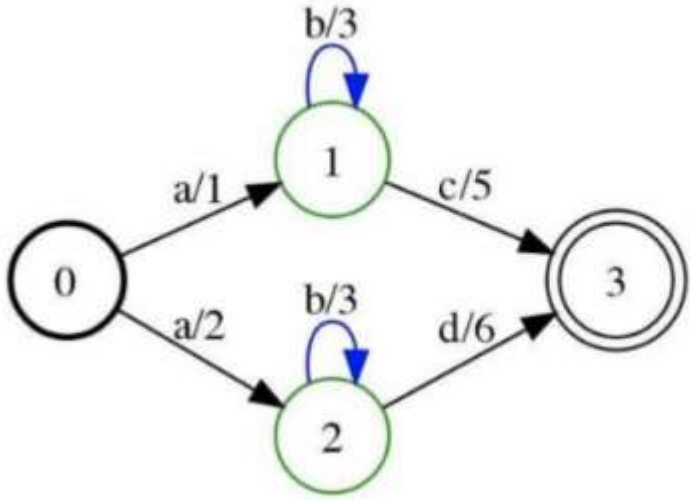


Figure: Original WFSA over Tropical Semiring

- Now consider $((1, 0), (2, 1))$.
- State "1" and "2" has residual weights 0 and 1 respectively.
- Compute total weight of given input "b":
  - $w(e_3) = 0 \otimes 3 = 0 + 3 = 3$
  - $w(e_4) = 1 \otimes 3 = 1 + 3 = 4$
  - $w(e_3) \oplus w(e_4) = min(3, 4) = 3$
- This is the new arc weight $w(e_2') = 3$

Determinization: Worked Example

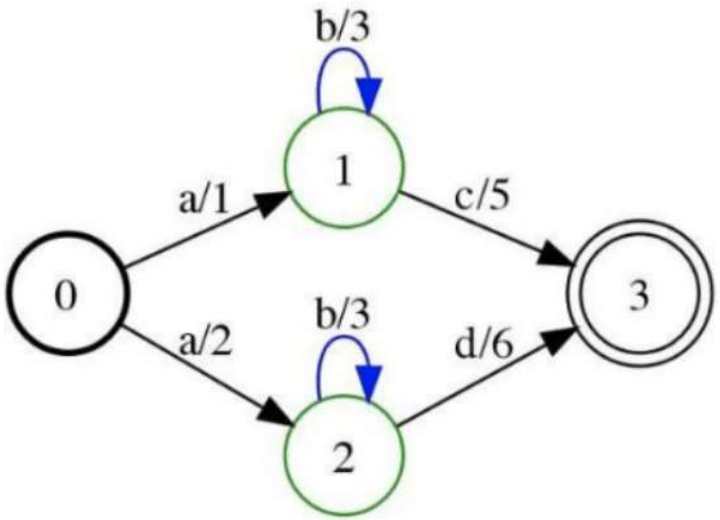


Figure: Original WFSA over Tropical Semiring

- Start by computing residual weights:
  - $w_r(e_i) = w(e_j')^{-1} \otimes w(p[e_i]) \otimes w(e_i)$
  - $w_r(e_3) = -3 + 0 + 3 = 0$
  - $w_r(e_4) = -3 + 1 + 3 = 1$
  - $n[e_3] =$"1" and $n[e_4] =$"2"
- Group using a new notation:
  - $((1, 0), (2, 1))$, the new state
- Then the next transition is:
  - $\{((1, 0), (2, 1)), b, 3, ((1, 0), (2, 1))\}$

Determinization: Worked Example

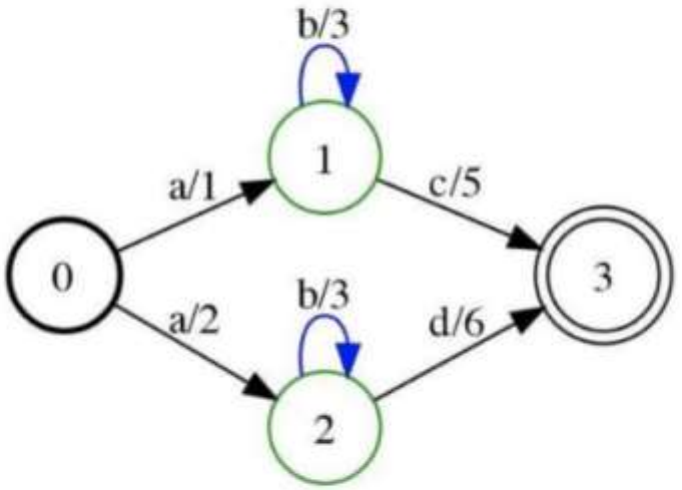


Figure: Original WFSA over Tropical Semiring

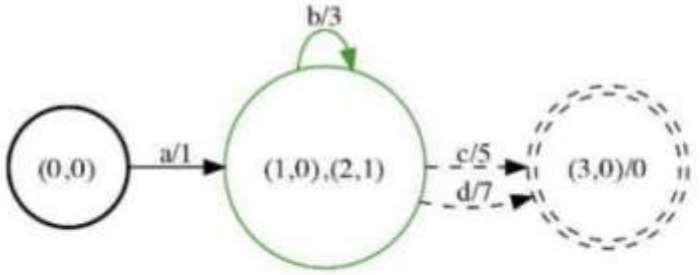


Figure: After second update

Determinization: Worked Example

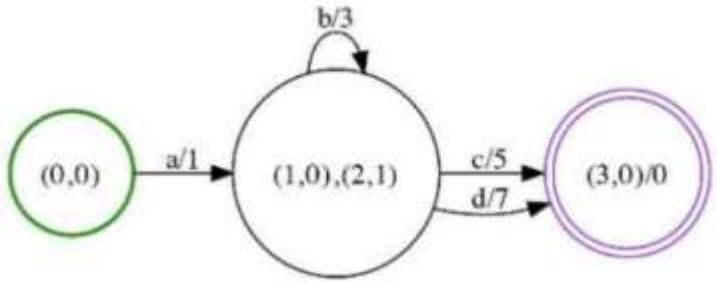
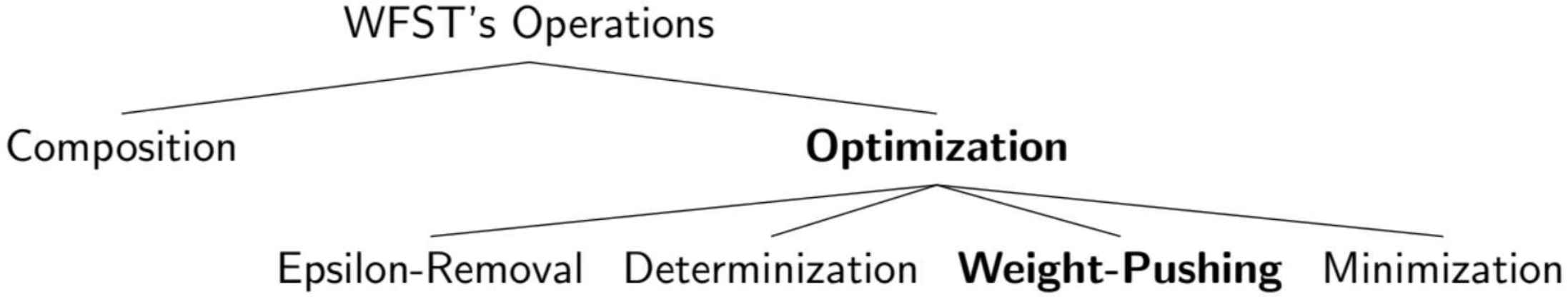


Figure: Determinized WFSA

- Some issues with weighted determinization:
  - May not terminate for some transducers
  - May not finish for some inputs, even if they are valid
  - However, all FSAs are determinizable


WFST's Operations
Composition
Optimization
Epsilon-Removal
Determinization
Weight-Pushing
Minimization

Introduction to Weight-Pushing

Moves the weights distributed over all the paths to the initial states, without changing the initial WFST/WFSA

Example: Given a WFST over a Tropical Semiring

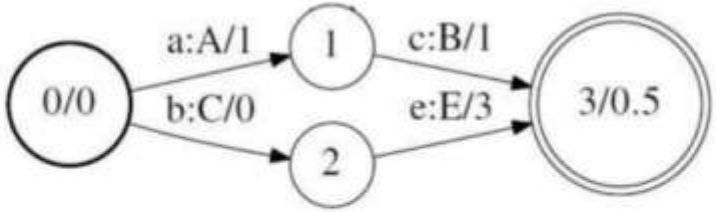


Figure: Original WFST, T

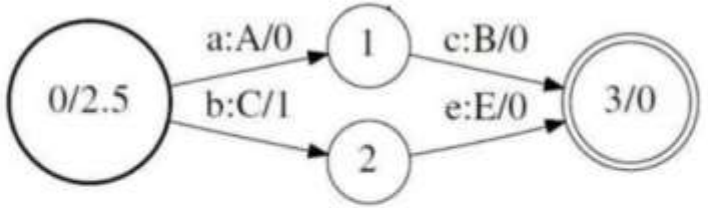


Figure: Weight-Pushed WFST in Tropical Semiring, $push(T)$

Rationale of Weight-Pushing

- Many sequence recognition or transduction problems reduces to finding the minimal cost path.
- As the unpromising paths can be eliminated in the early stage of the search, and therefore reducing the total search time.

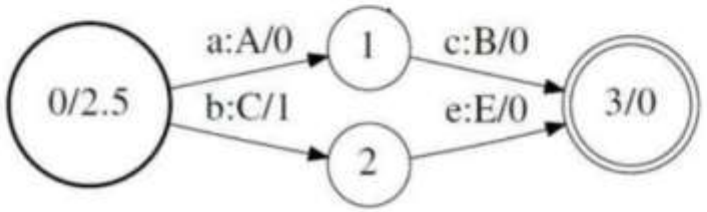


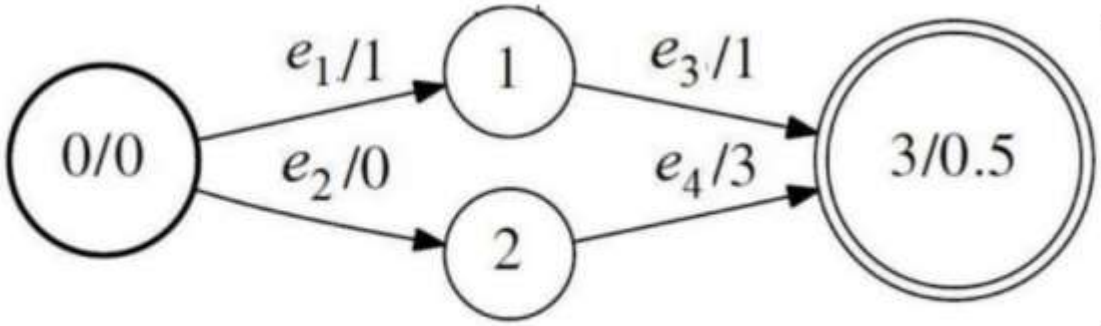
Figure: Weight-Pushed WFST in Tropical Semiring, $push(T)$

Description Of Weight-Pushing Algorithm

Given a WFST $T : (\Sigma, \Delta, Q, E, I, F, \lambda, \rho)$
over semiring $S = (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

1. Compute *potential* of each state, given by
   - $V[q] = \bigoplus_{\pi \in \Pi(q,F)} w(\pi) \otimes \rho(n[\pi])$
   - Where $\Pi(q, F)$ is the set of paths originating from state q
   - In the event of an infinite loop, employ the *k-closed* property
2. Re-weighting the **transition weights, initial weights and final weights** in the following way:
   - $\forall e \in E$ s.t $V[p[e]] \neq \bar{0}$, $w[e] \leftarrow V[p[e]]^{-1} \otimes w[e] \otimes V[n[e]]$
   - $\forall q \in I$, $\lambda(q) \leftarrow \lambda(q) \otimes V[q]$
   - $\forall q \in F$, $\rho(q) \leftarrow V[q]^{-1} \otimes \rho(q)$

Worked Example (T over Tropical Semiring)



(a) Original WFST.

e_1	("0",–,–, 1, "1")
e_2	("0",–,–, 0, "2")
e_3	("1",–,–, 1, "3")
e_4	("2",–,–, 3, "3")

First compute *potential* of each state:

- $V[q] = \bigoplus_{\pi \in \Pi(q,F)} w(\pi) \otimes \rho(n[\pi])$

Worked Example (T over Tropical Semiring)

Recall

Potential of state: $V[q] = \bigoplus_{\pi \in \Pi(q,F)} w(\pi) \otimes \rho(n[\pi])$

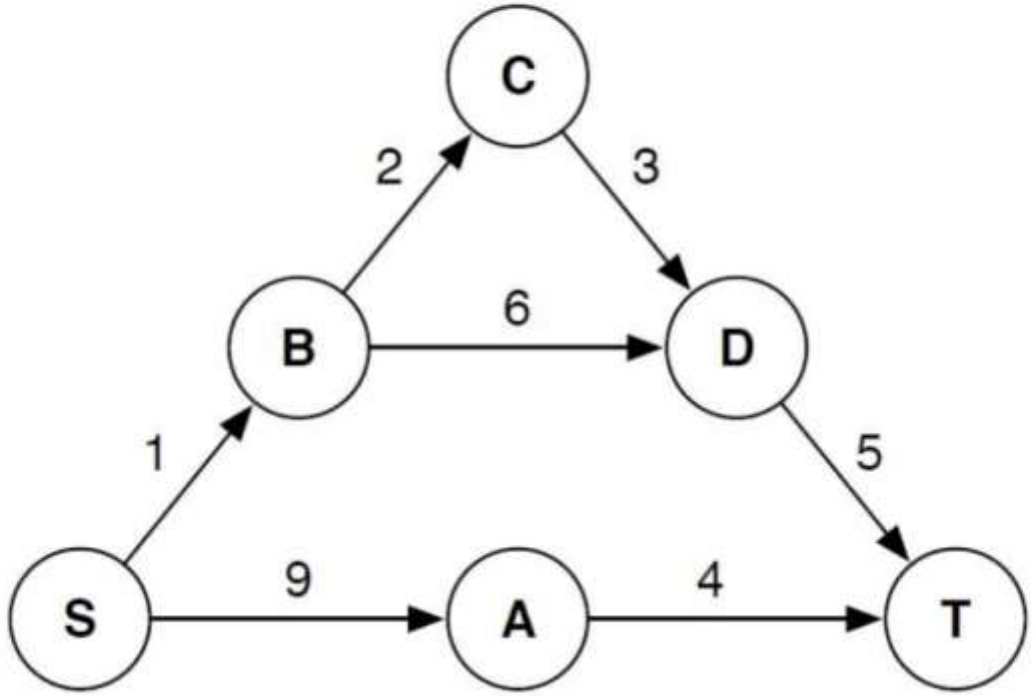
Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Step 1.1: State "0" has 2 paths going to the final state "3"

- $\pi_1 = e_1, e_3$ and $\pi_2 = e_2, e_4$
- $w(\pi_1) \otimes \rho(n[\pi_1]) = (1 \otimes 1) \otimes (0.5) = 1 + 1 + 0.5 = 2.5$
- $w(\pi_2) \otimes \rho(n[\pi_2]) = (0 \otimes 3) \otimes (0.5) = 0 + 3 + 0.5 = 3.5$
- Using the potential formula
  - $V[0] = 2.5 \oplus 3.5 = min(2.5, 3.5) = 2.5$

Based on this Semiring, what are we doing?

Recall: Shortest Path from S to T



Step 1, Compute:

- $9 + 4 = 13$
- $1 + 6 + 5 = 12$
- $1 + 2 + 3 + 5 = 11$

Step 2, Then:

- $min\{13, 12, 11\} = 11$

Worked Example (T over Tropical Semiring)

Recall

Potential of state: $V[q] = \bigoplus_{\pi \in \Pi(q,F)} w(\pi) \otimes \rho(n[\pi])$

Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Step 1.2: State "1" has 1 path going to the final state "3"

- $\pi_3 = e_3$
- $w(\pi_3) \otimes \rho(n[\pi_3]) = (1) \otimes (0.5) = 1 + 0.5 = 1.5$
- Using the potential formula
  - $V[1] = 1.5 \oplus \bar{0} = min(1.5, \infty) = 1.5$

Worked Example (T over Tropical Semiring)

Recall

Potential of state: $V[q] = \bigoplus_{\pi \in \Pi(q,F)} w(\pi) \otimes \rho(n[\pi])$

Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

Step 1.4: State "3" is the final state, initialize $V[3] = \rho(3) = 0.5$

Hence we obtain all the potentials in the transducer:

- $V[0] = 2.5$
- $V[1] = 1.5$
- $V[2] = 3.5$
- $V[3] = 0.5$

Worked Example (T over Tropical Semiring)

Recall: Re-weighting

1. $\forall e \in E$ s.t $V[p[e]] \neq \bar{0}$, $w[e] \leftarrow V[p[e]]^{-1} \otimes w[e] \otimes V[n[e]]$
2. $\forall q \in I$, $\lambda(q) \leftarrow \lambda(q) \otimes V[q]$
3. $\forall q \in F$, $\rho(q) \leftarrow V[q]^{-1} \otimes \rho(q)$
4. Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- $V[0] = 2.5$
- $V[1] = 1.5$
- $V[2] = 3.5$
- $V[3] = 0.5$

Step 2.1: Using (2) and (3), re-weight initial and final states "0" and "3"

- $\lambda(0) \leftarrow \lambda(0) \otimes V[0] = 0 \otimes 2.5 = 0 + 2.5 = 2.5$
- $\rho(3) \leftarrow V[3]^{-1} \otimes \rho(3) = V[3]^{-1} \otimes \ V[3] = \bar{1} = 0$

Worked Example (T over Tropical Semiring)

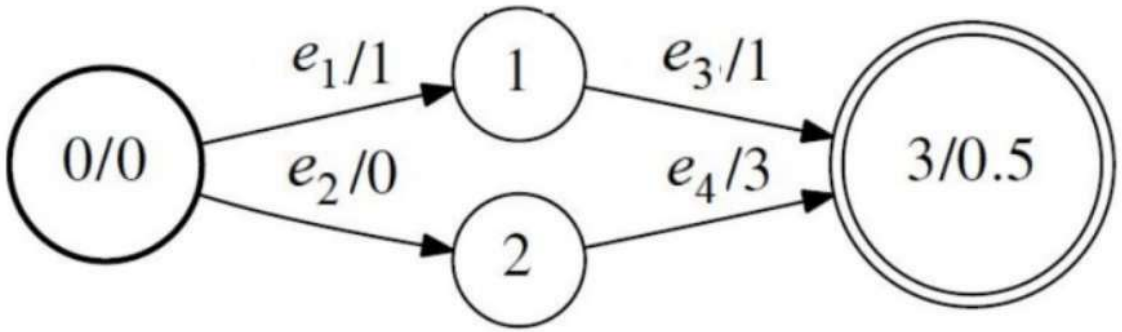


Figure: Before Update

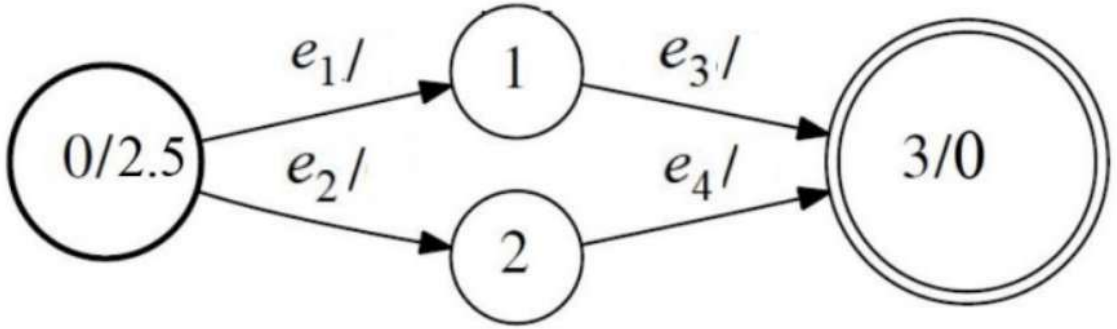


Figure: After Update

Worked Example (T over Tropical Semiring)

Recall: Re-weighting

1. $\forall e \in E$ s.t $V[p[e]] \neq \bar{0}$, $w[e] \leftarrow V[p[e]]^{-1} \otimes w[e] \otimes V[n[e]]$
2. $\forall q \in I$, $\lambda(q) \leftarrow \lambda(q) \otimes V[q]$
3. $\forall q \in F$, $\rho(q) \leftarrow V[q]^{-1} \otimes \rho(q)$
4. Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- $V[0] = 2.5$
- $V[1] = 1.5$
- $V[2] = 3.5$
- $V[3] = 0.5$

Step 2.3: Using (1), re-weight the transition $e_2 = (0, 0, 2)$

- $w[e_2] \leftarrow V[p[e_2]]^{-1} \otimes w[e_2] \otimes V[n[e_2]]$
- $V[0]^{-1} \otimes w[e_2] \otimes V[2] = -2.5 + 0 + 3.5 = 1$
- $w[e_2] \leftarrow 1$

Worked Example (T over Tropical Semiring)

Recall: Re-weighting

1. $\forall e \in E$ s.t $V[p[e]] \neq \bar{0}$, $w[e] \leftarrow V[p[e]]^{-1} \otimes w[e] \otimes V[n[e]]$
2. $\forall q \in I$, $\lambda(q) \leftarrow \lambda(q) \otimes V[q]$
3. $\forall q \in F$, $\rho(q) \leftarrow V[q]^{-1} \otimes \rho(q)$
4. Tropical Semiring: $(\mathbb{R}, min, +, \infty, 0) \equiv (\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$

- $V[0] = 2.5$
- $V[1] = 1.5$
- $V[2] = 3.5$
- $V[3] = 0.5$

Step 2.5: Using (1), re-weight the transition $e_4 = (2, 3, 3)$

- $w[e_4] \leftarrow V[p[e_4]]^{-1} \otimes w[e_4] \otimes V[n[e_4]]$
- $V[2]^{-1} \otimes w[e_4] \otimes V[3] = -3.5 + 3 + 0.5 = 0$
- $w[e_4] \leftarrow 0$

Worked Example (T over Tropical Semiring)

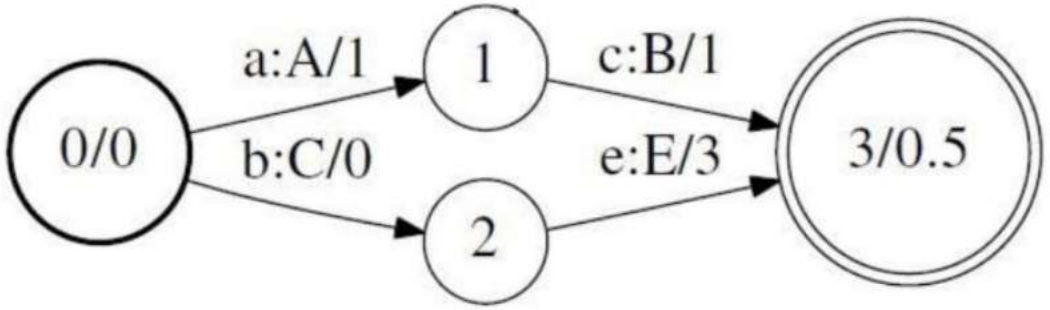


Figure: Before Weight-Pushing, T

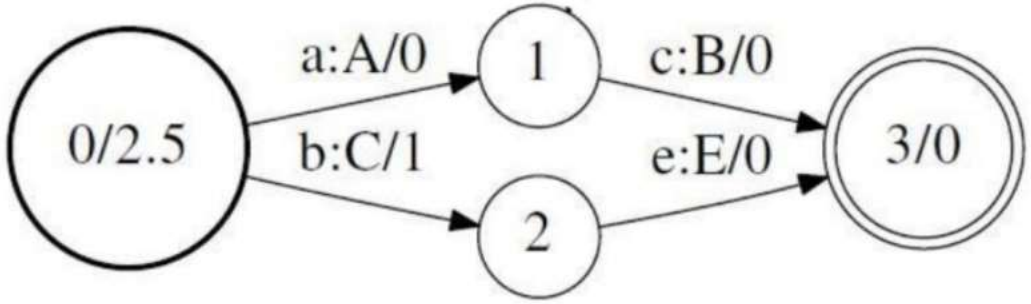
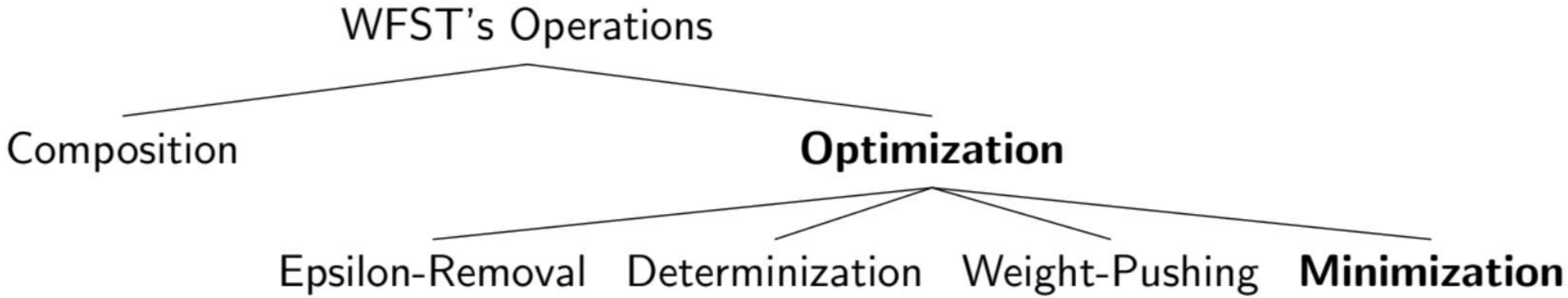


Figure: After Weight-Pushing, $push(T)$ over a Tropical Semiring

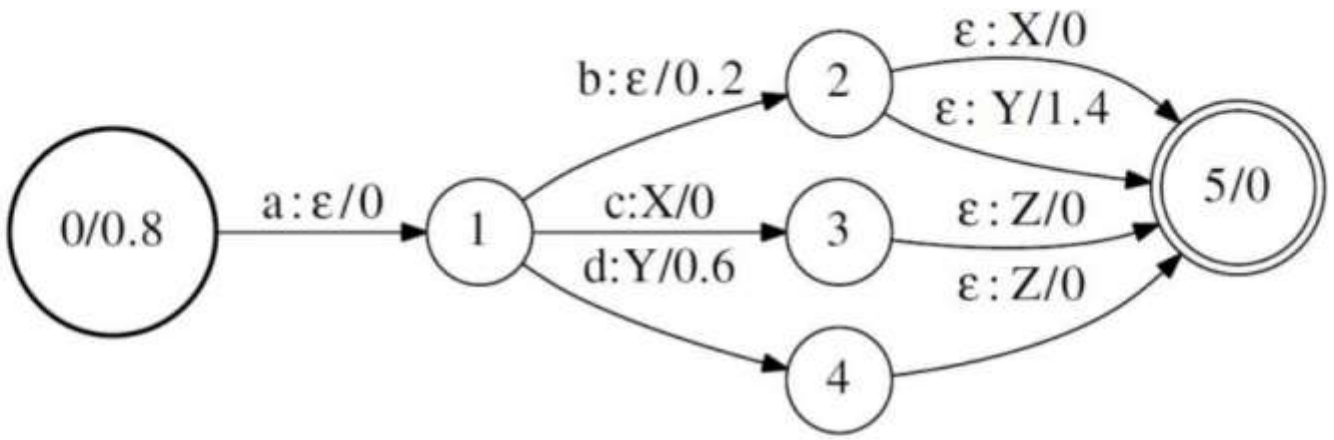

WFST's Operations
Composition
Optimization
Epsilon-Removal
Determinization
Weight-Pushing
Minimization

Introduction to Minimization

- Minimize the number of states for any DFA. Hence, given a transducer T, usually a determinization operation is performed, resulting in det(T)
- First, push weights and output labels to the initial states in the WFST
- Then, minimize the WFST using a classical minimization algorithm assuming that the triplet input:output/weight on each transition as one single label.

Minimization Algorithm I

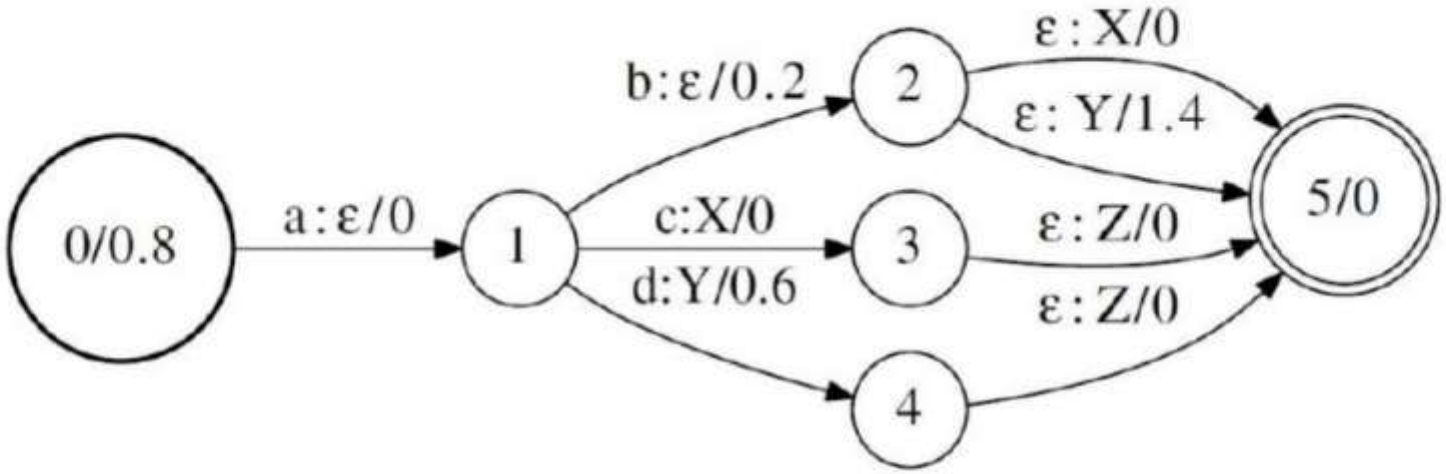
For example, given a determinized WFST $det(T)$:



(a) Weight-pushed WFST: push(det(T)), where det(T)

Minimization Algorithm II

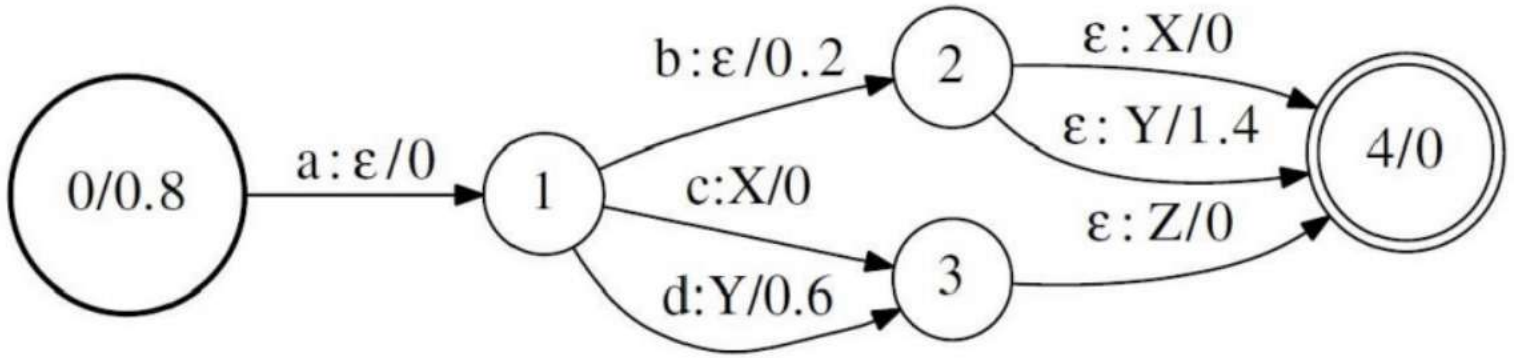
Perform weight-pushing on $det(T)$:



to obtain $push(det(T))$

Minimization Algorithm III

Finally, apply any minimization algorithm on $push(det(T))$:



And obtain $min(push(det(T)))$

Rationale for Minimization

- Hopcrofts algorithm is an algorithm for efficiently minimizing DFAs. If WFST is acyclic, a more efficient algorithm, Revuzs algorithm, is used.
- By reducing the number of states, the search paths are greatly reduced, shortening search time.

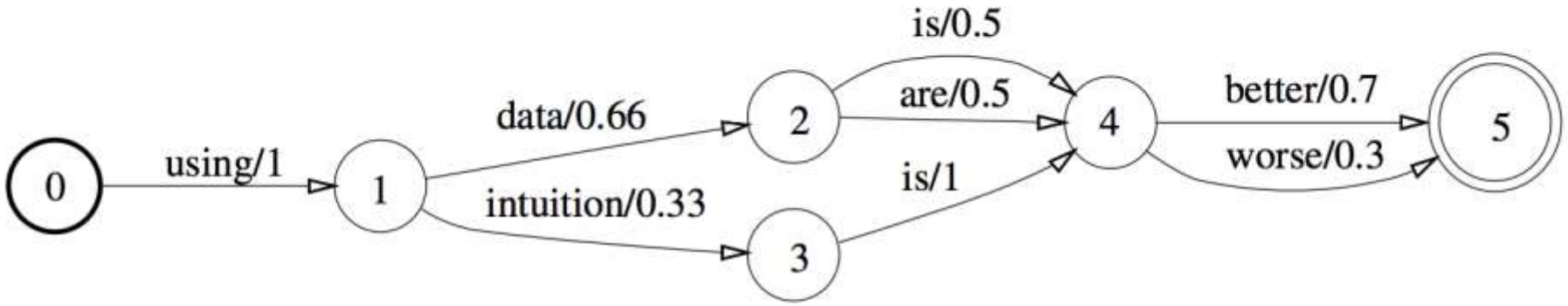
Some Applications of WFSTs

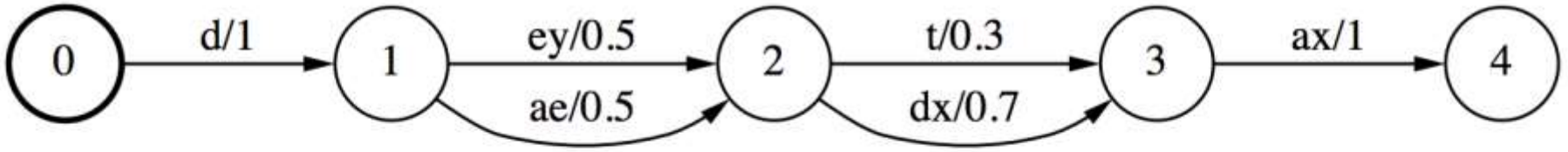
- Speech Recognition
- Optical Character Recognition
- Text Classification
- Machine translation

第三课 语言模型与解码对齐

- 知识点1：解码与对齐（DECODING, ALIGNMENT）：从孤立词识别到连接词/词序列之CONNECTED WORD RECOGNITION与TIME ALIGNMENT
- 知识点2： WFST介绍、WFST基本操作COMPOSITION/DETERMINISATION/MINIMISATION
- 知识点3： WFST在ASR中的应用：HCLG、基于WFST的BEAM SEARCH

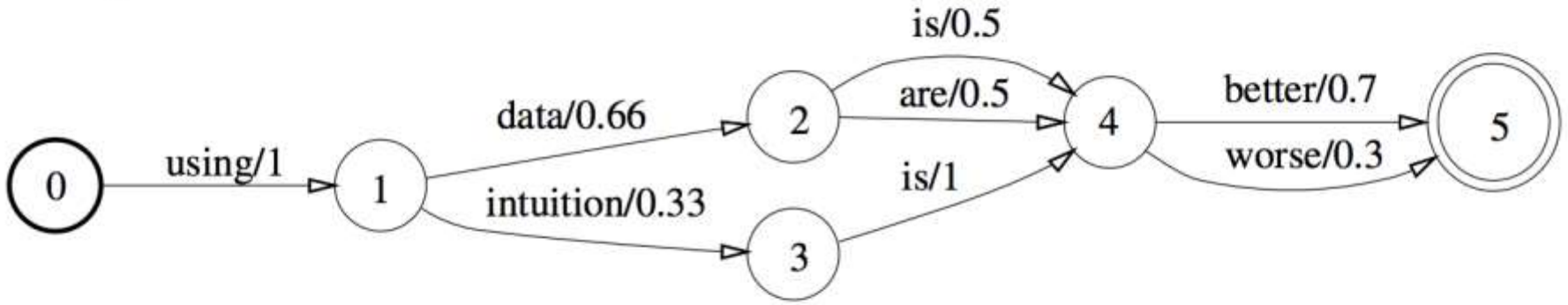
Weighted Finite State Acceptors



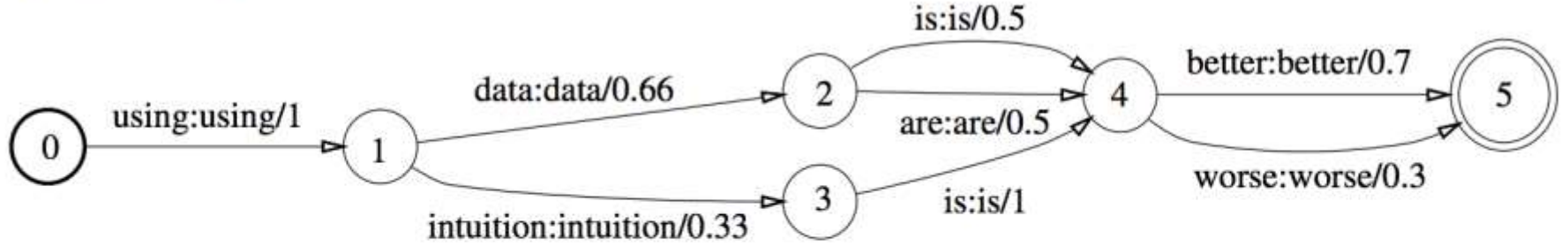


Weighted Finite State Transducers

Acceptor

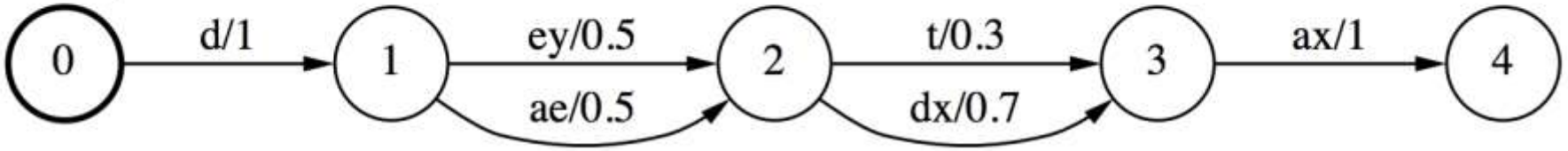


Transducer

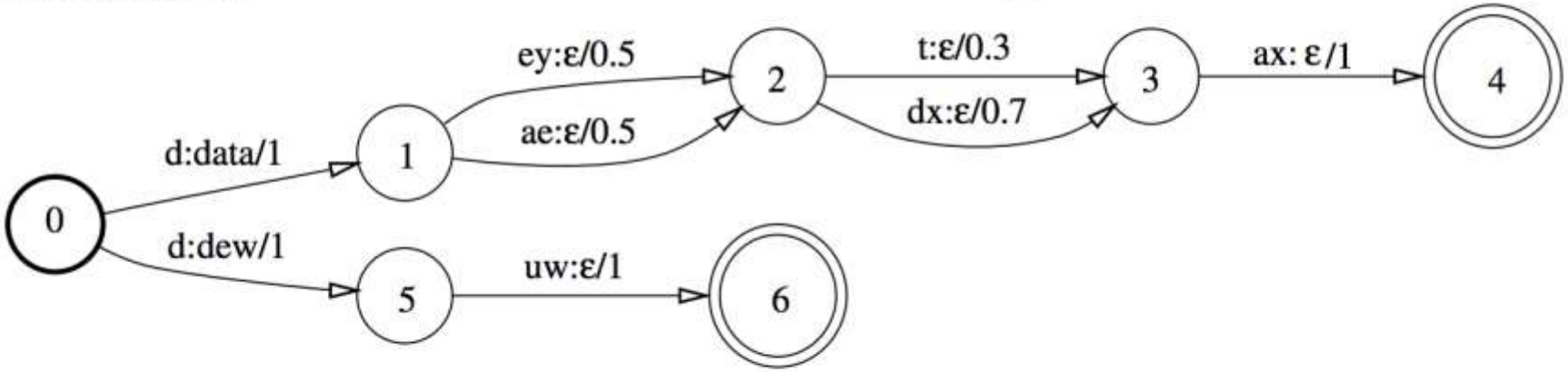


Weighted Finite State Transducers

Acceptor



Transducer



WFST Algorithms

Composition Combine transducers at different levels. For example if G is a finite state grammar and L is a pronunciation dictionary then $L \circ G$ transduces a phone string to word strings allowed by the grammar

Determinisation Ensure that each state has no more than a single output transition for a given input label

Minimisation transforms a transducer to an equivalent transducer with the fewest possible states and transitions

Applying WFSTs to speech recognition

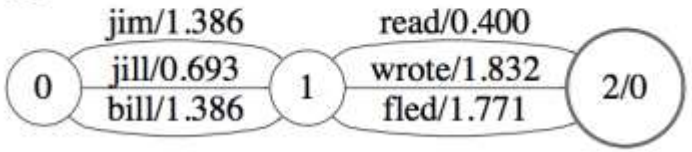
- Represent the following components as WFSTs

	transducer	input sequence	output sequence
G	word-level grammar	words	words
L	pronunciation lexicon	phones	words
C	context-dependency	CD phones	phones
H	HMM	HMM states	CD phones

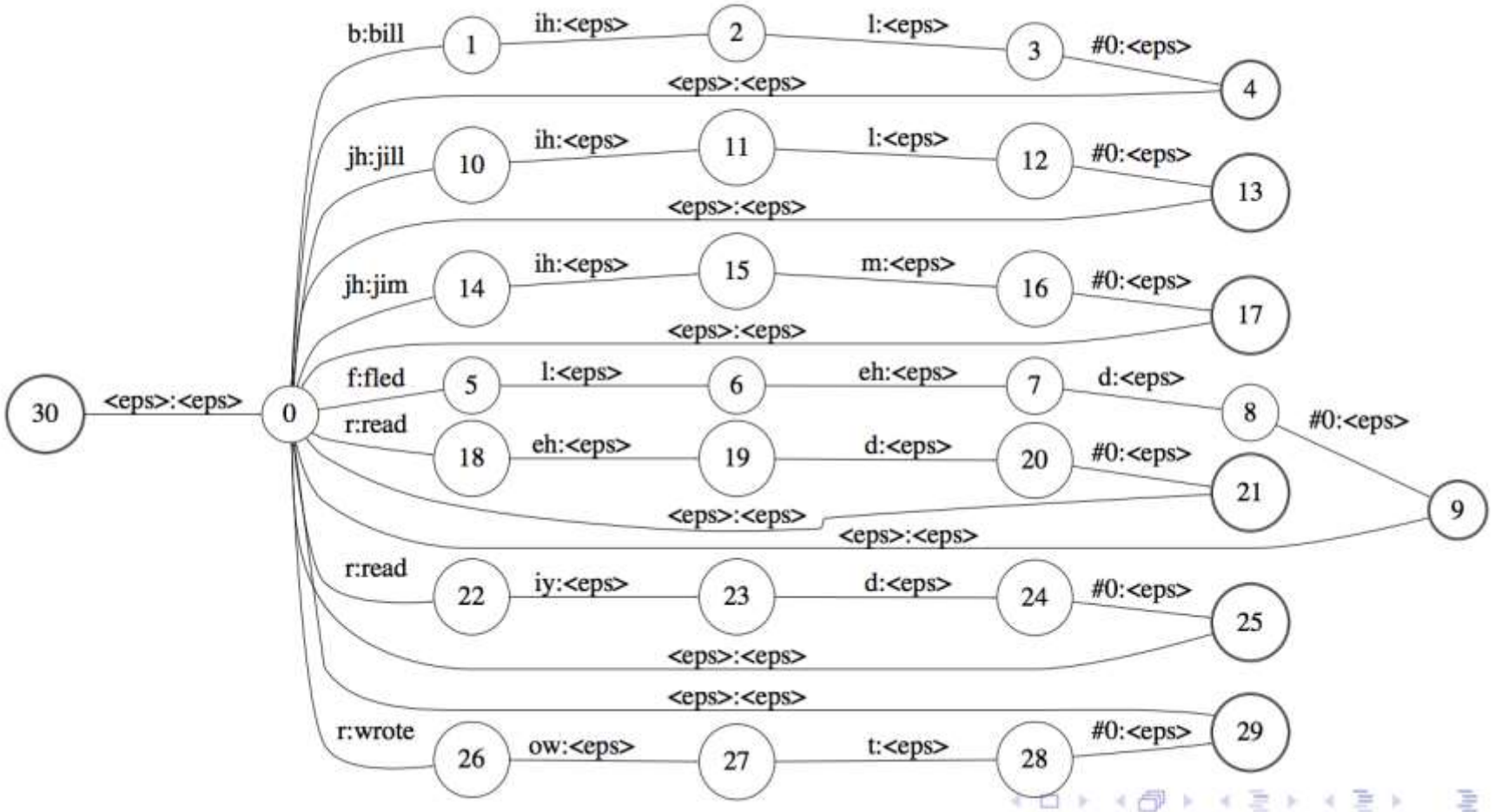
- Composing L and G results in a transducer $L \circ G$ that maps a phone sequence to a word sequence
- $H \circ C \circ L \circ G$ results in a transducer that maps from HMM states to a word sequence

L, G

G

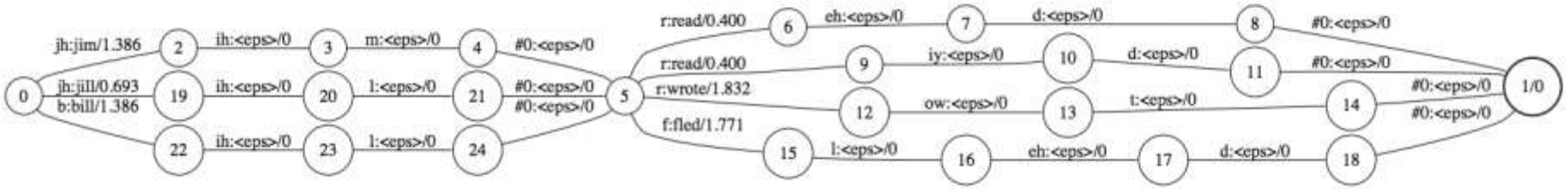

jim/1.386
jill/0.693
bill/1.386
read/0.400
wrote/1.832
fled/1.771
0
1
2/0

L

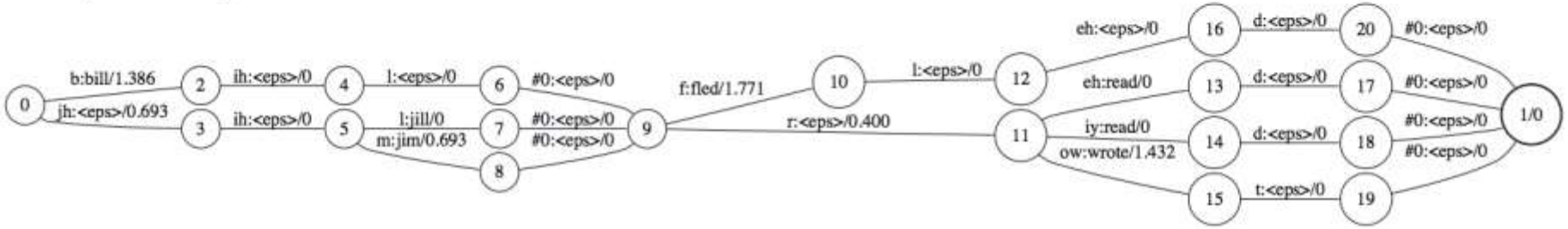

30
<eps>:<eps>
0
b:bill
1
ih:<eps>
2
l:<eps>
3
#0:<eps>
4
<eps>:<eps>
jh:jill
10
ih:<eps>
11
l:<eps>
12
#0:<eps>
13
<eps>:<eps>
jh:jim
14
ih:<eps>
15
m:<eps>
16
#0:<eps>
17
<eps>:<eps>
f:fled
5
l:<eps>
6
eh:<eps>
7
d:<eps>
8
#0:<eps>
r:read
18
eh:<eps>
19
d:<eps>
20
#0:<eps>
21
9
<eps>:<eps>
<eps>:<eps>
r:read
22
iy:<eps>
23
d:<eps>
24
#0:<eps>
25
<eps>:<eps>
<eps>:<eps>
r:wrote
26
ow:<eps>
27
t:<eps>
28
#0:<eps>
29

$L \circ G$, $\det(L \circ G)$, $\min(\det(L \circ G))$

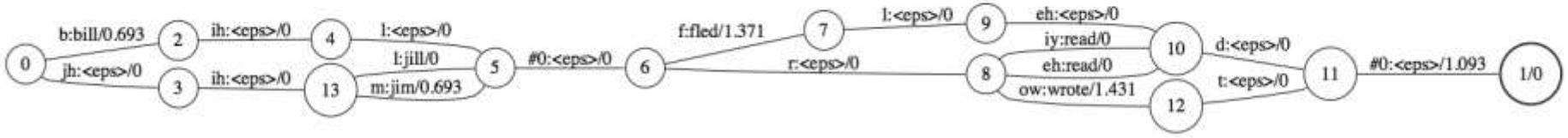
$L \circ G$



$\det(L \circ G)$

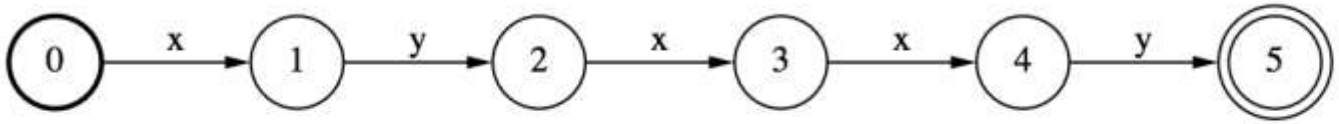


$\min(\det(L \circ G))$

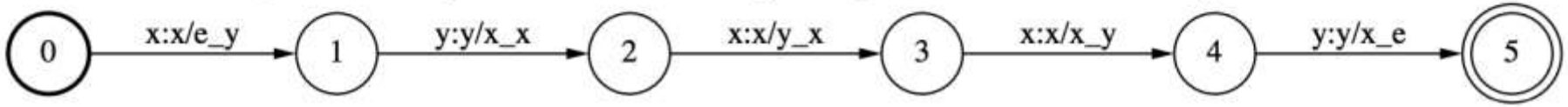


Context dependency transducer C

Context-independent "string"



Context-dependency transducer (weights not shown)



(x/e_y – x with left context e (start/end) and right context y)

Decoding using WFSTs

- We can represent the HMM acoustic model, pronunciation lexicon and n-gram language model as four transducers: H, C, L, G
- Combining the transducers gives an overall "decoding graph" for our ASR system – but minimisation and determination means it is much smaller than naively combining the transducers
- But it is important in which order the algorithms are combined otherwise the transducers may "blow-up" – basically after each composition, first determinise then minimise
- In Kaldi, ignoring one or two details

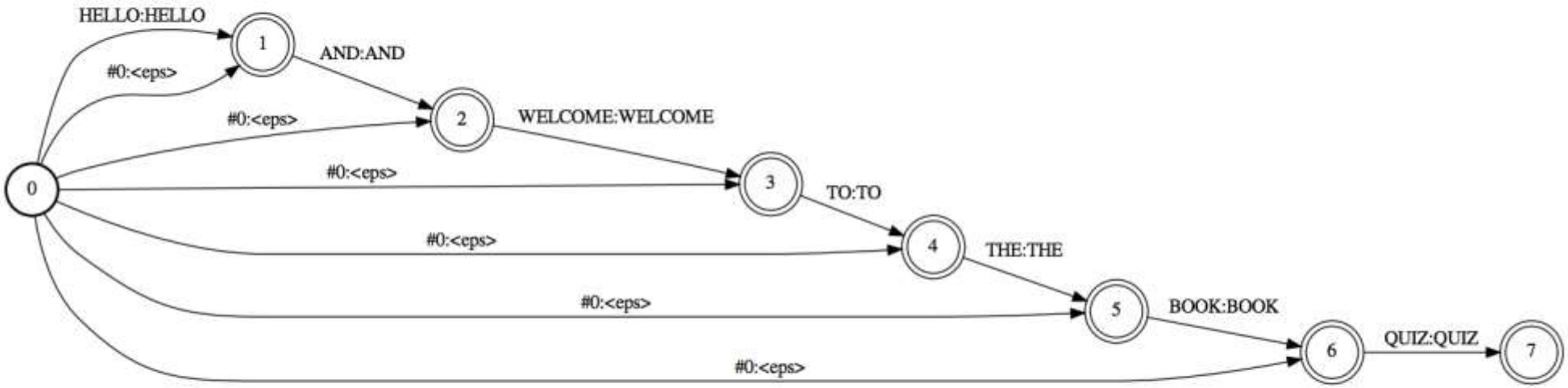
$$HCLG = \min(\det(H \circ \min(\det(C \circ \min(\det(L \circ G))))))$$

Alignment

- Alignment is the task of matching a recording to a transcript
- In many circumstances the available transcript differs from a verbatim transcript: for example, captions/subtitles for a TV programme may not include every word spoken, or may include paraphrasing
- Performing alignment using such transcripts is of great practical use
  - time-aligning subtitles to the broadcast
  - using the data for speech recognition training (*lightly supervised training*)
- In lightly supervised training we need to use the alignment to identify reliable labels and learn from them – without also learning from unreliable labels, or past mistakes

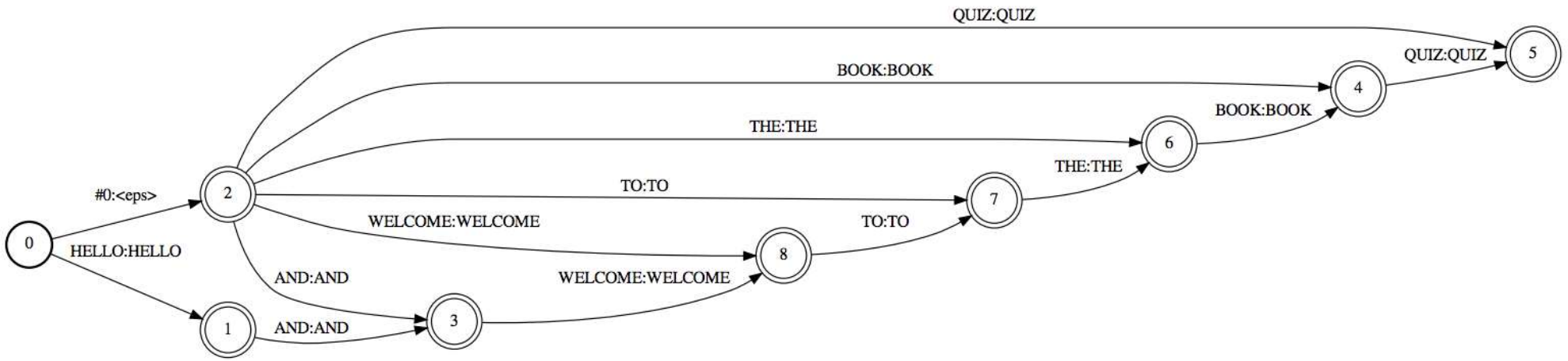
Alignment with WFSTs

A *G* transducer that allows any substring of the original captions – known as a *factor transducer*



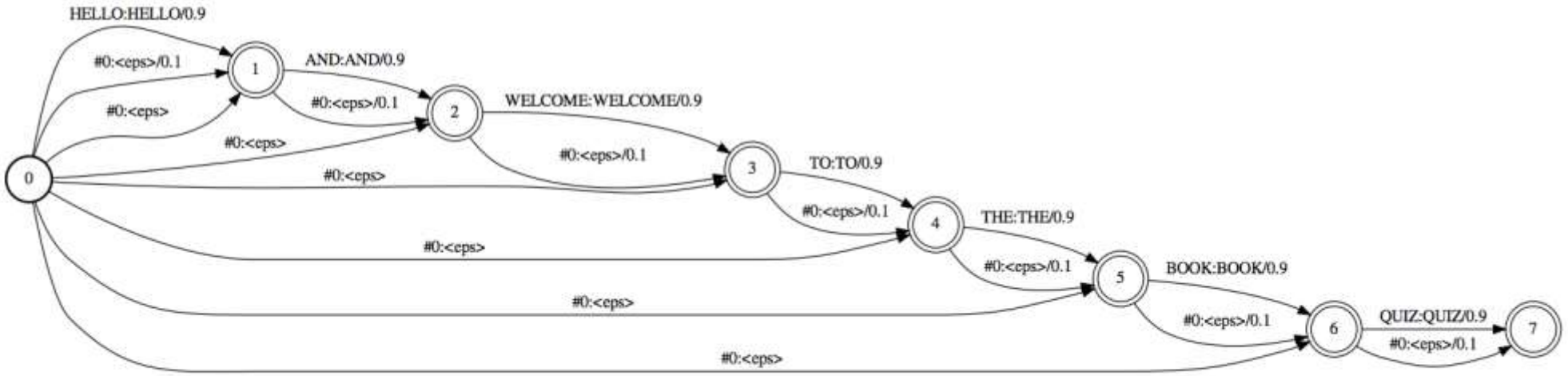
Alignment with WFSTs

A determinized version of the G transducer



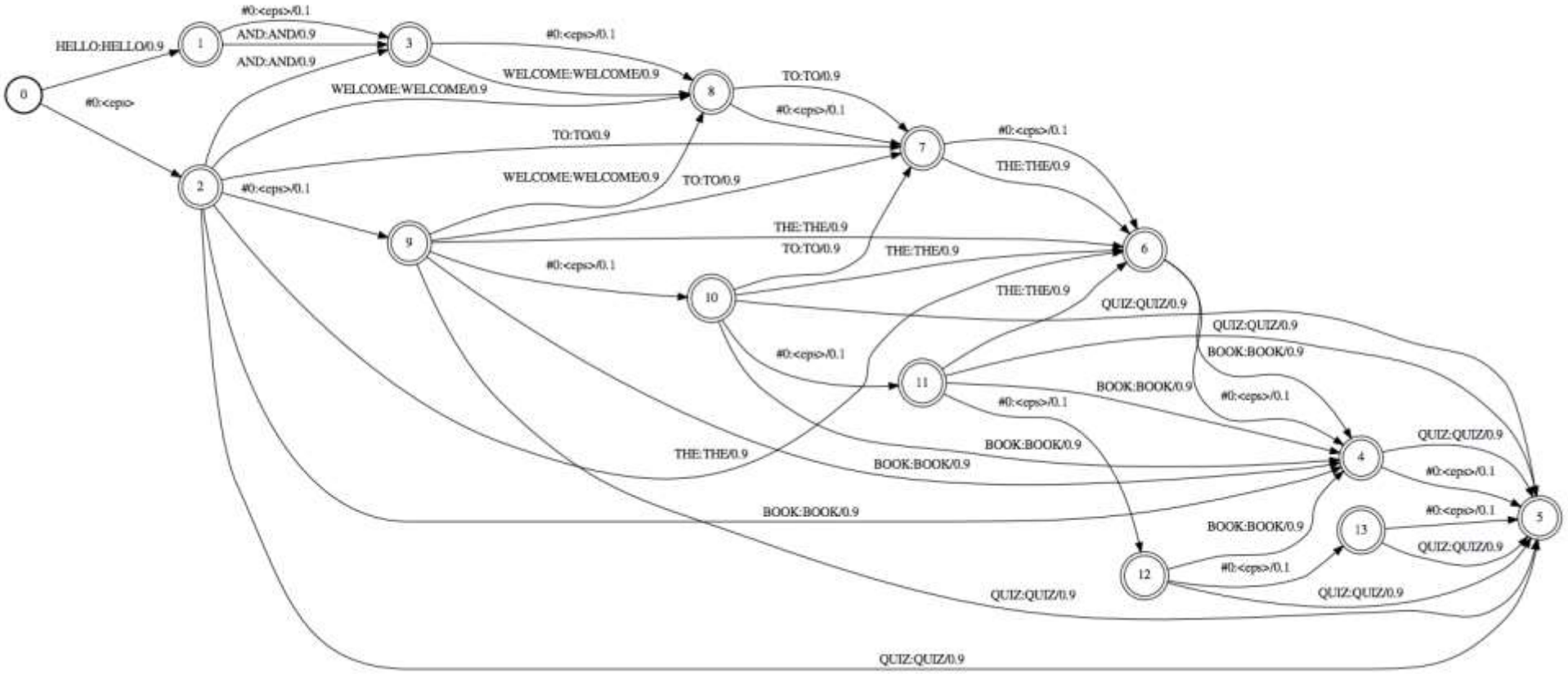
Alignment with WFSTs

What about when word appears in the captions that was not actually spoken?
Alter the design to be robust to this by allowing deletions (at a cost)



Alignment with WFSTs

A determinized version



The complete alignment process

1. Decode with a factor-transducer for the each programme
2. Align the output to the original captions
3. Re-segment the data, to potentially include missed speech
4. Decode again with utterance-specific factor transducers, allowing word-skips

Summary

- Search (decoding) in ASR involves finding the correct word sequence given a sample recording
- Weighted finite state transducer (WFST) framework – provides a well-justified way to combine models at different levels
- WFST algorithms - composition, determinisation, minimisation
- Kaldi represents a speech recogniser as an HCLG transducer – combining 4 transducers to map from HMM states to word sequences
- WFSTs provide a way to represent various problems in speech recognition, eg alignment

博客推荐（KALDI）

- 编译 & 命令行工具
  - HTTPS://WWW.TWBLOGS.NET/A/5B8B2E102B717718832DE330/ZH-CN
- 加权有限状态转换器
  - HTTP://VSOODA.GITHUB.IO/2016/08/28/WFST/#SPEECH-RECOGNITION-TRANSDUCERS
- OPENFST
  - HTTP://WWW.OPENFST.ORG/TWIKI/BIN/VIEW/FST/WEBHOME
- KALDI GUIDE
  - HTTP://DANIELPOVEY.COM/FILES/LECTURE4.PDF

THANKS