HMMs and WFSTs

Lecture 4

CS 753

Instructor: Preethi Jyothi

Recap: HMMs for Acoustic Modeling

- What are the three fundamental problems related to HMMs?

- 1. What is the forward algorithm? What is it used to compute?

  Computing Likelihood: Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

- 2. What is the Viterbi algorithm? What is it used to compute?

  Decoding: Given as input an HMM $\lambda = (A, B)$ and a sequence of observations $O = o_1, o_2, ..., o_T$, find the most probable sequence of states $Q = q_1 q_2 q_3 \ldots q_T$.

- 3. What is the Baum-Welch algorithm? What does it compute?

  Learning: Given an observation sequence O and the set of possible states in the HMM, learn the HMM parameters A and B.

Monophone HMMs? Not good enough. Need Triphones.

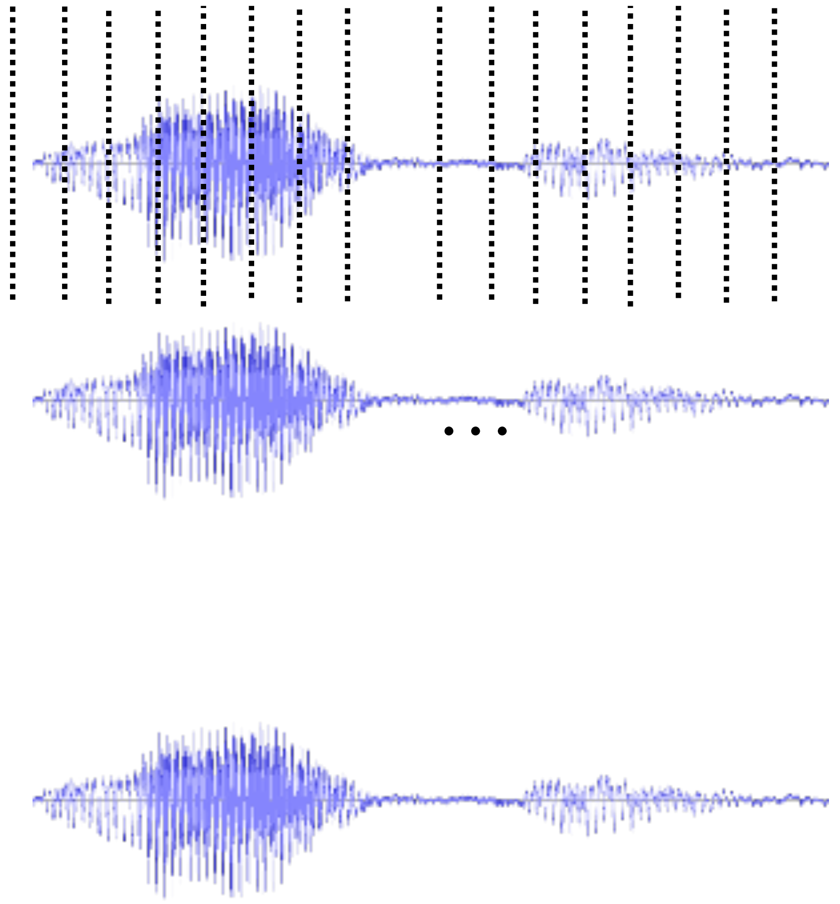
- A phone is affected by its phonetic context.
  - E.g. Coarticulation: Production of a speech sound is affected by adjacent speech sounds. "soon" vs. "seat". "ten" vs. "tenth".

- For modelling, use phones in context instead of monophones. E.g. diphones or triphones.

- Triphones are commonly used in ASR systems. Phone p with left context l and right context r is written as "l-p+r"
  - "hello world" → sil-h+eh h-eh+l eh-l+ow l-ow+w ow-w-er w-er+l er-l+d l-d+sil

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate AM parameters $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all triphone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-triphone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Estimate LM parameters β using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Test

$\mathbf{O} = O_1, \ldots, O_T$

$$W^* = \arg\max_W P(W|O) = \arg\max_W P_\theta(O|W) P_\beta(W)$$

Compute using Viterbi algorithm

Search all possible state sequences arising from all word sequences most likely to have generated $\mathbf{O}$

Computationally infeasible for continuous speech! WFSTs to the rescue, to make this more tractable.

What are Weighted Finite State Transducers (WFSTs)?

(Weighted) Automaton

- Accepts a subset of strings (over an alphabet), and rejects the rest
  - Mathematically, specified by $L \subseteq \Sigma^*$ or equivalently $f : \Sigma^* \rightarrow \{0,1\}$
- Weighted: outputs a “weight” as well (e.g., probability)
  - $f : \Sigma^* \rightarrow W$
- Transducer: outputs another string (over possibly another alphabet)
  - $f : \Sigma^* \times \Delta^* \rightarrow W$

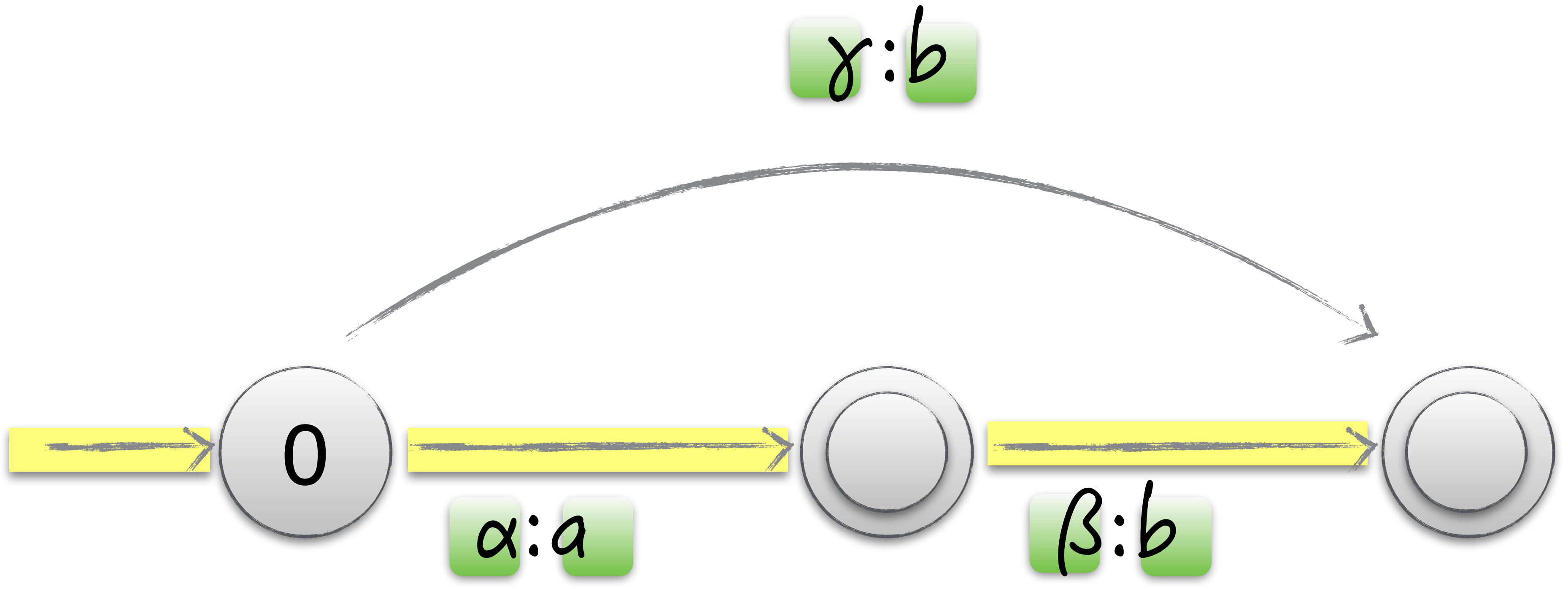
(Weighted) Finite State Automaton

- Functions that can be implemented using a machine which:
  - reads the string one symbol at a time
  - has a fixed amount of memory: so, at any moment, the machine can be in only one of finitely many states, irrespective of the length of the input string
- Allows efficient algorithms to reason about the machine
  - e.g., output string with maximum weight for input $\alpha\beta\gamma$

Why WFSTs?

- Powerful enough to (reasonably) model processes in language, speech, computational biology and other machine learning applications
  - Simpler WFSTs can be combined to create complex WFSTs, e.g., speech recognition systems
- If using WFST models, efficient algorithms available to train the models and to make inferences
  - Toolkits that don't have domain specific dependencies

Structure: Finite State Transducer (FST)

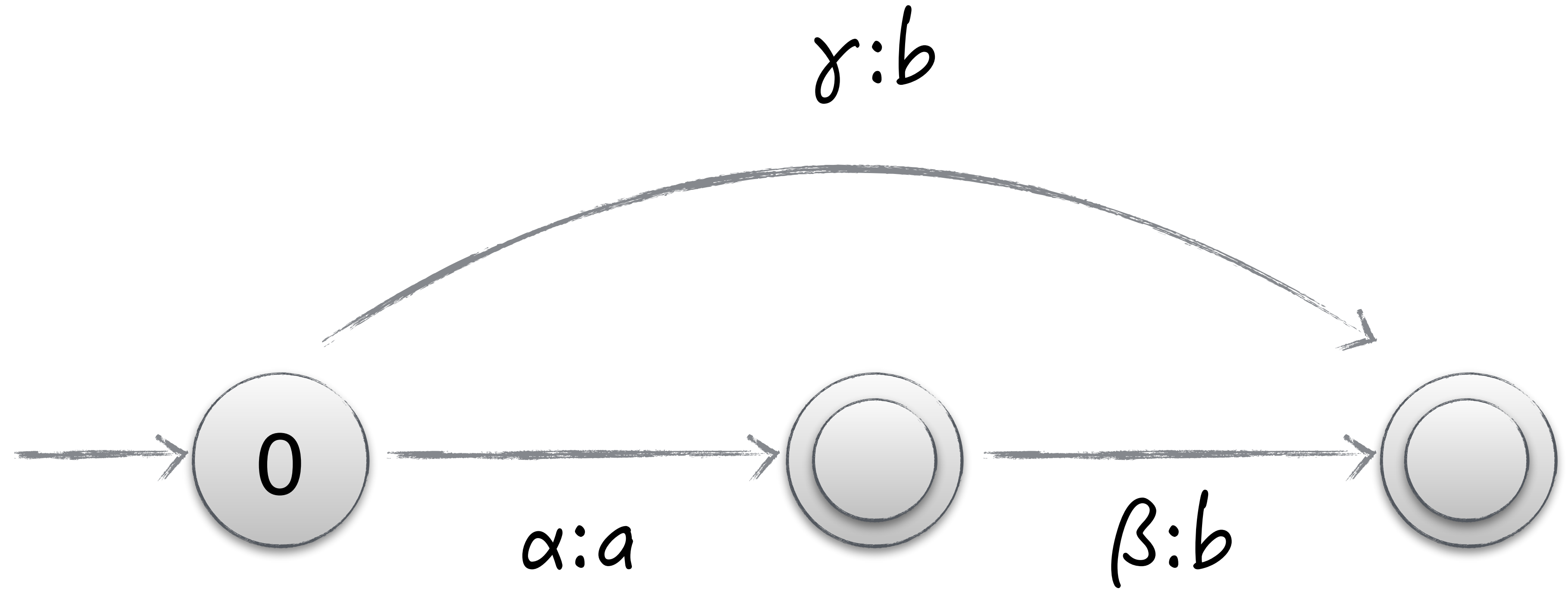


Elements of an FST

- States
- Start state (0)
- Final states (1 & 2)
- Arcs (transitions)
- Input symbols (from alphabet Σ)
- Output symbols (from alphabet Δ)

FST maps input strings to output strings

Path



- A successful "path" → Sequence of transitions from the start state to any final state

- Input label of a path → Concatenation of input labels on arcs. Similarly for output label of a path.

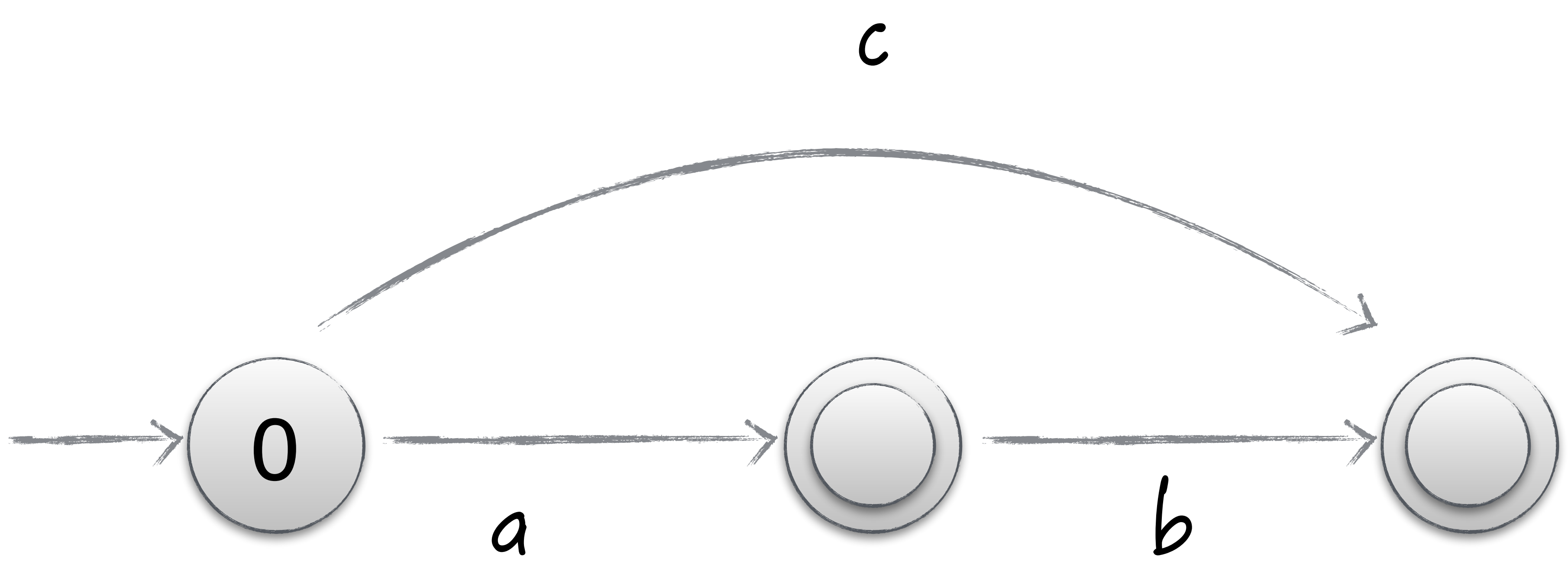
FSAs and FSTs

- Finite state acceptors (FSAs)
  - Each transition has a source & destination state, and a label
  - FSA accepts a set of strings, $L \subseteq \Sigma^*$

FSA can be thought of as a special kind of FST

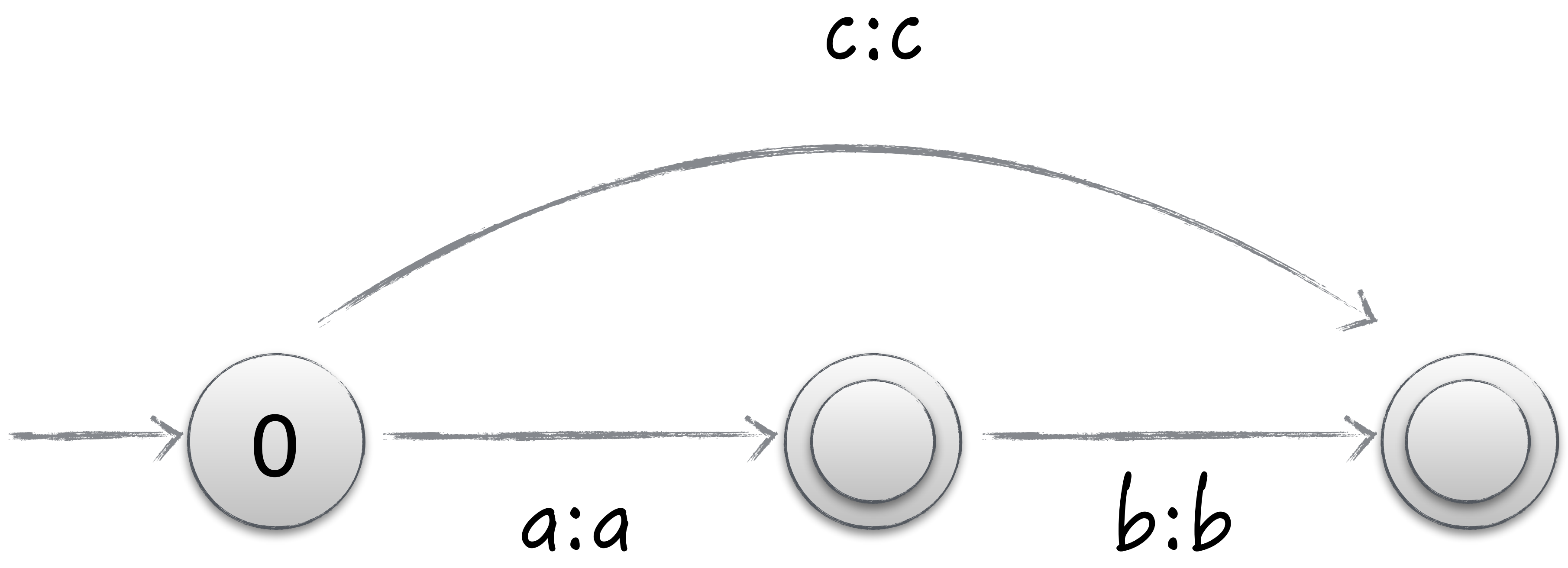
- Finite state transducers (FSTs)
  - Each transition has a source & destination state, an input label and an output label
  - FST represents a relation, $R \subseteq \Sigma^* \times \Delta^*$

Example of an FSA



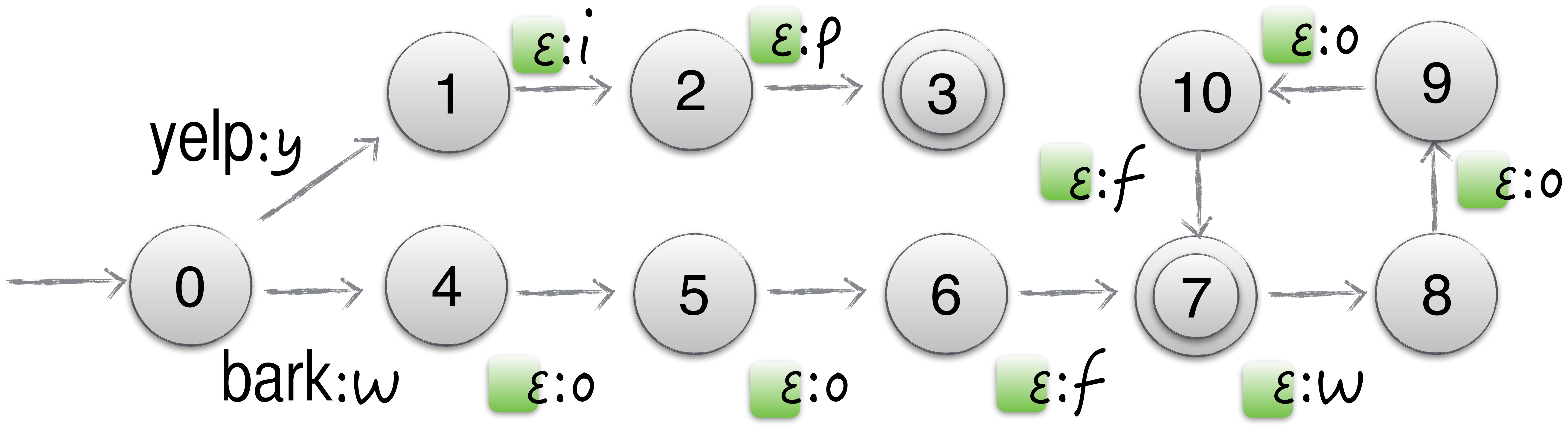
Accepts strings
{c, a, ab}

Equivalent FST



Accepts strings
{c, a, ab}
and outputs identical strings
{c, a, ab}

Barking dog FST



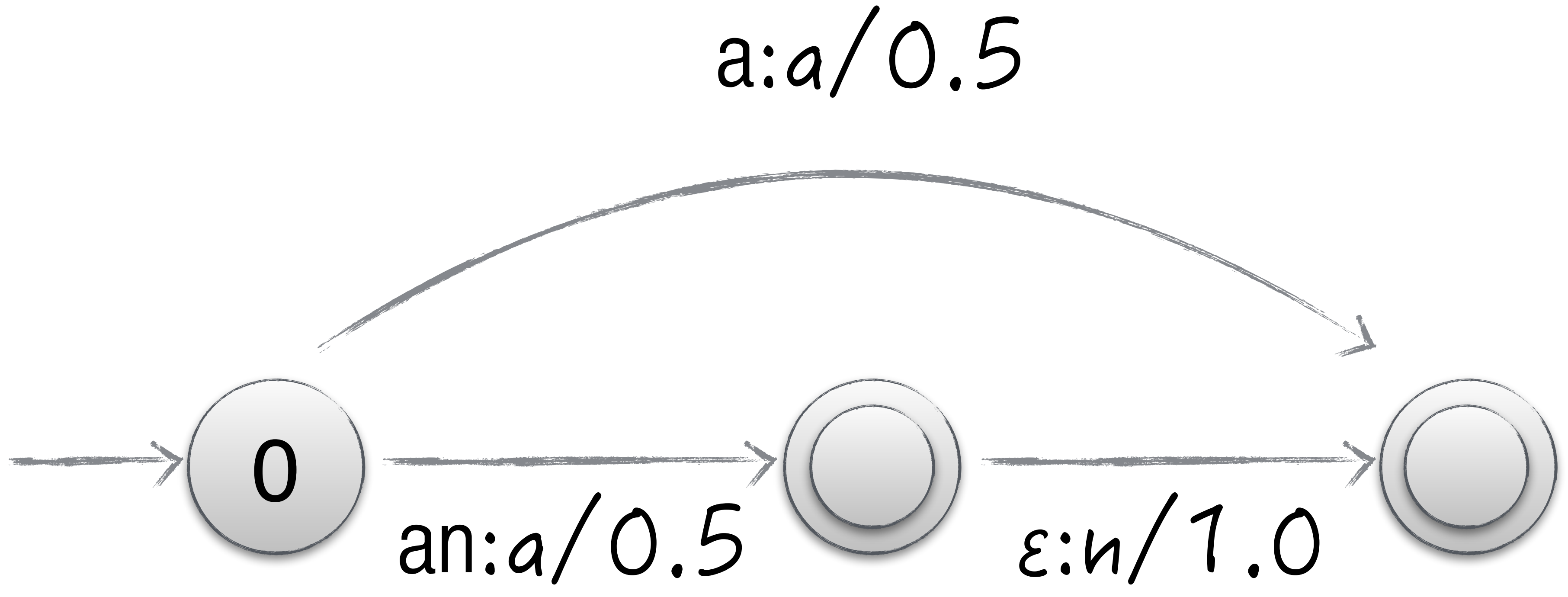
Σ = { yelp, bark }, Δ = { a, …, z }

yelp → y i p. bark → w o o f | w o o f w o o f |…

Special symbol, ε (epsilon) : allows to make a move without consuming an input symbol

or without producing an output symbol

Weighted Path

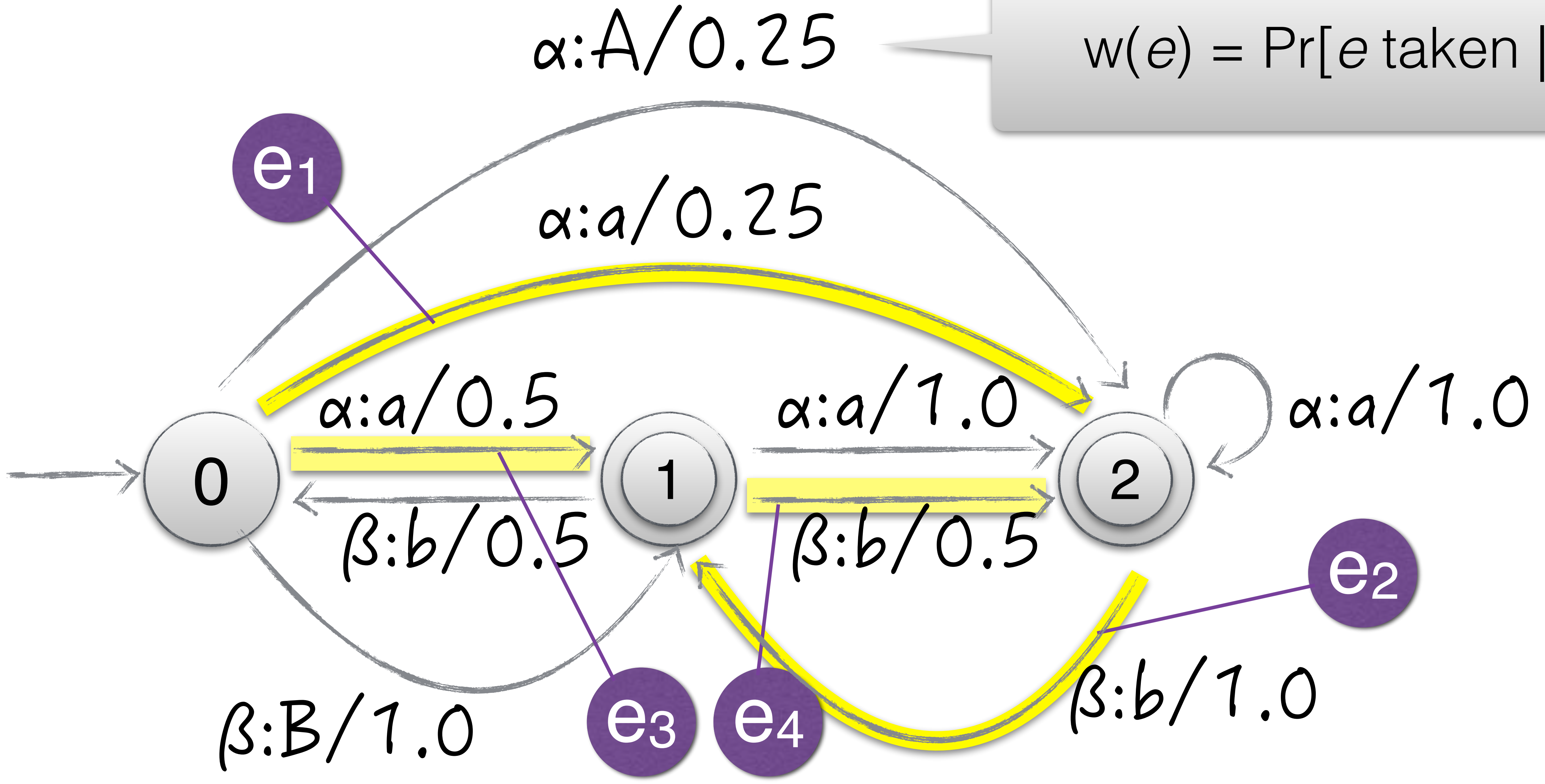


- "Weights" can be probabilities, negative log-likelihoods, or any cost function representing the cost incurred in mapping an input sequence to an output sequence

- How are the weights accumulated along a path?

Weighted Path: Probabilistic FST

$w(e) = \Pr[e \text{ taken} \mid \text{state}=0, \text{in-symbol}=\alpha]$



$\pi_1 = e_1 e_2$
$\pi_2 = e_3 e_4$

- $T(\alpha\beta, ab) = \Pr[\text{output}=ab, \text{accepts} \mid \text{input}=\alpha\beta, \text{start}]$

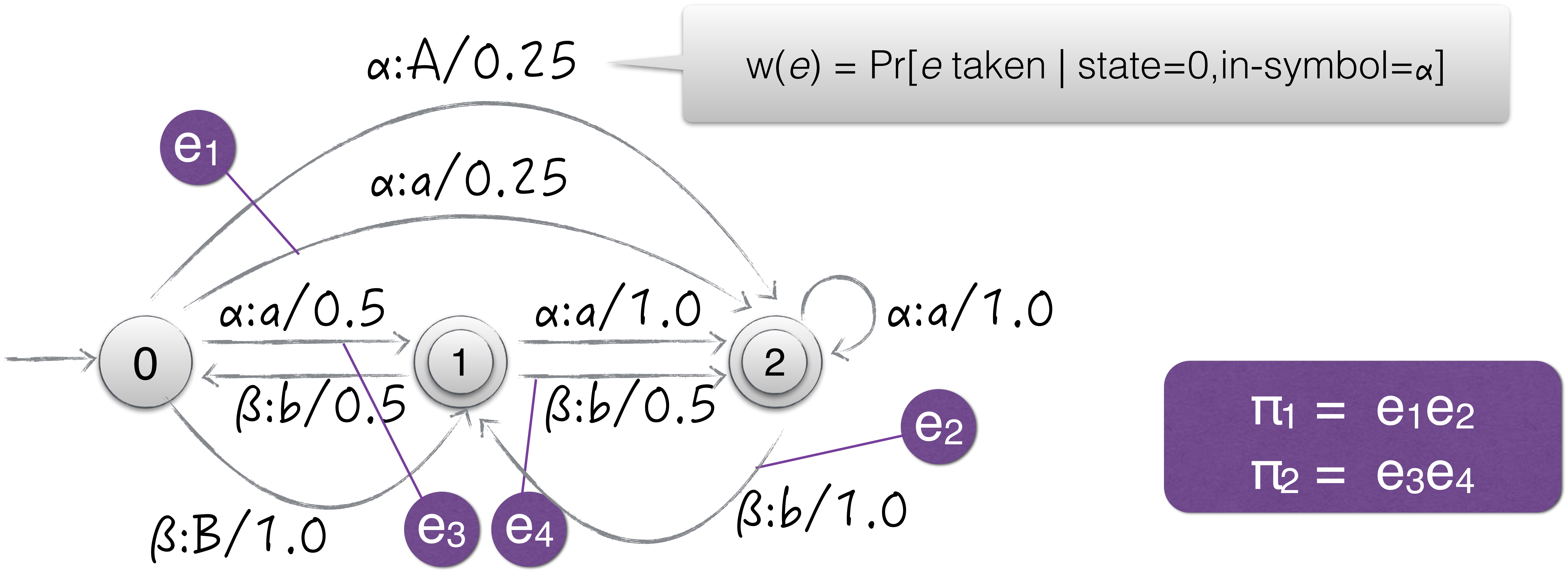
$= 0.25 + 0.25 = 0.5$

$= \Pr[\pi_1 \mid \text{input}=\alpha\beta, \text{start}] + \Pr[\pi_2 \mid \text{input}=\alpha\beta, \text{start}]$

$= \Pr[e_1 \mid \text{input}=\alpha\beta, \text{start}] \times \Pr[e_2 \mid \text{input}=\alpha\beta, \text{start}, e_1]$
$= \Pr[e_1 \mid \text{state}=0, \text{in-symb}=\alpha] \times \Pr[e_2 \mid \text{state}=2, \text{in-symb}=\beta]$
$= w(e_1) \times w(e_2) = 0.25 \times 1.0 = 0.25$

$= w(e3) \times w(e4) = 0.5 \times 0.5 = 0.25$

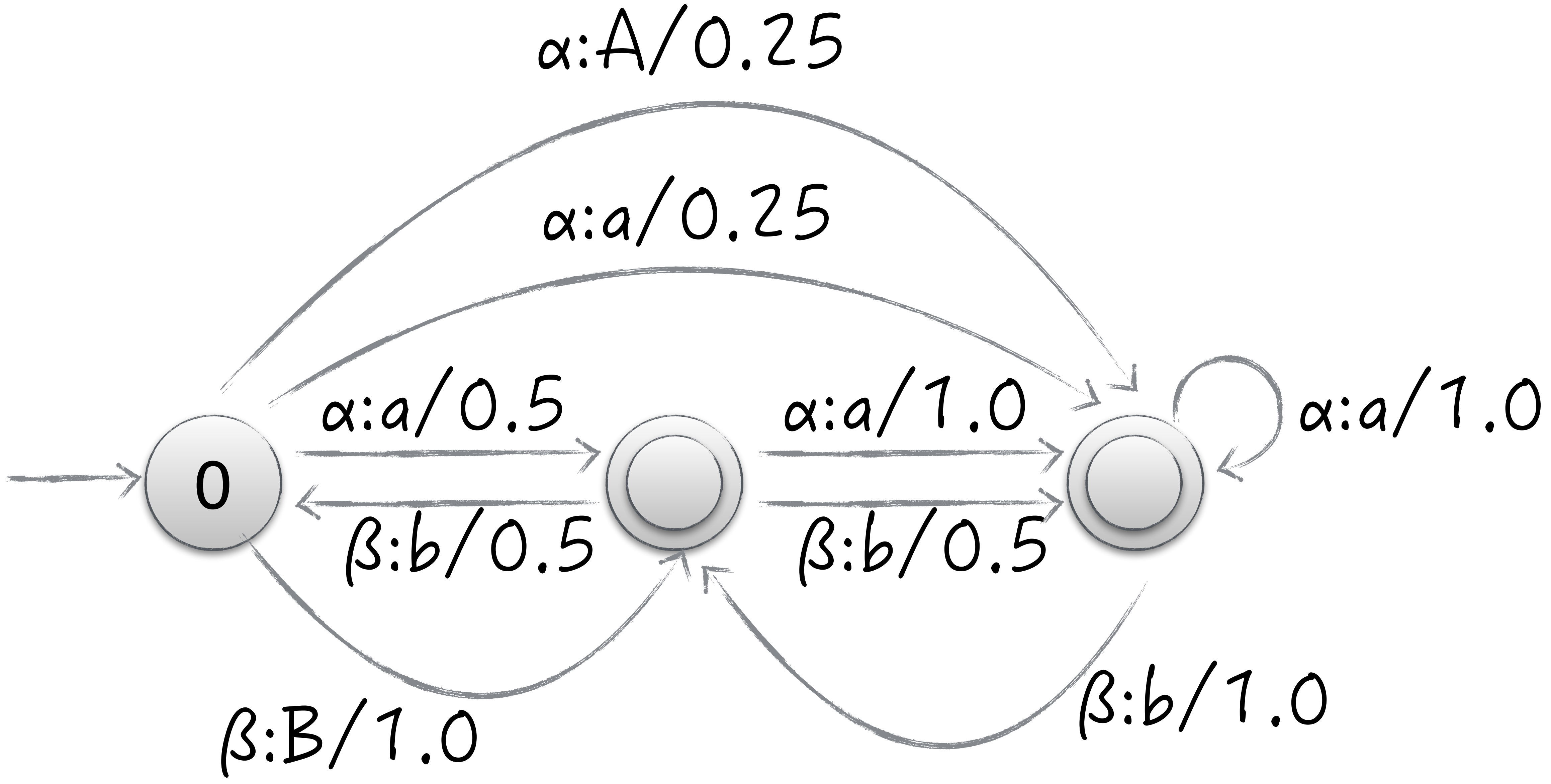
Weighted Path: Probabilistic FST



- $T(\alpha\beta, ab)$ = Pr[output=ab, accepts | input=$\alpha\beta$, start]
  = Pr[π_1 | input=$\alpha\beta$, start] + Pr[π_2 | input=$\alpha\beta$, start]

- More generally, $T(x,y) = \Sigma_{\pi \in P(x,y)} \; \Pi_{e \in \pi} \; w(e)$
  where $P(x,y)$ is the set of all accepting paths with input x and output y

Weighted Path



- But not all WFSTs are probabilistic FSTs
- Weight is often a “score” and maybe accumulated differently
- But helpful to retain some basic algebraic properties of weights: abstracted as semirings

Semirings

A semiring is a set of values associated with two operations $\oplus$ and $\otimes$, along with their identity values $\bar{0}$ and $\bar{1}$

Weight assigned to an input/output pair

$$T(x,y) = \bigoplus_{\pi \in P(x,y)} \bigotimes_{e \in \pi} w(e)$$

where P(x,y) is the set of all accepting paths with input x, output y

(generalizing the weight function for a probabilistic FST)

Semirings

Some popular semirings [M02]

SEMIRING	SET	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Boolean	{F,T}	$\vee$	$\wedge$	F	T
Real	$\mathbb{R}_+$	+	×	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{\log}$	+	+∞	0
Tropical	$\mathbb{R} \cup \{-\infty, +\infty\}$	min	+	+∞	0

Is there

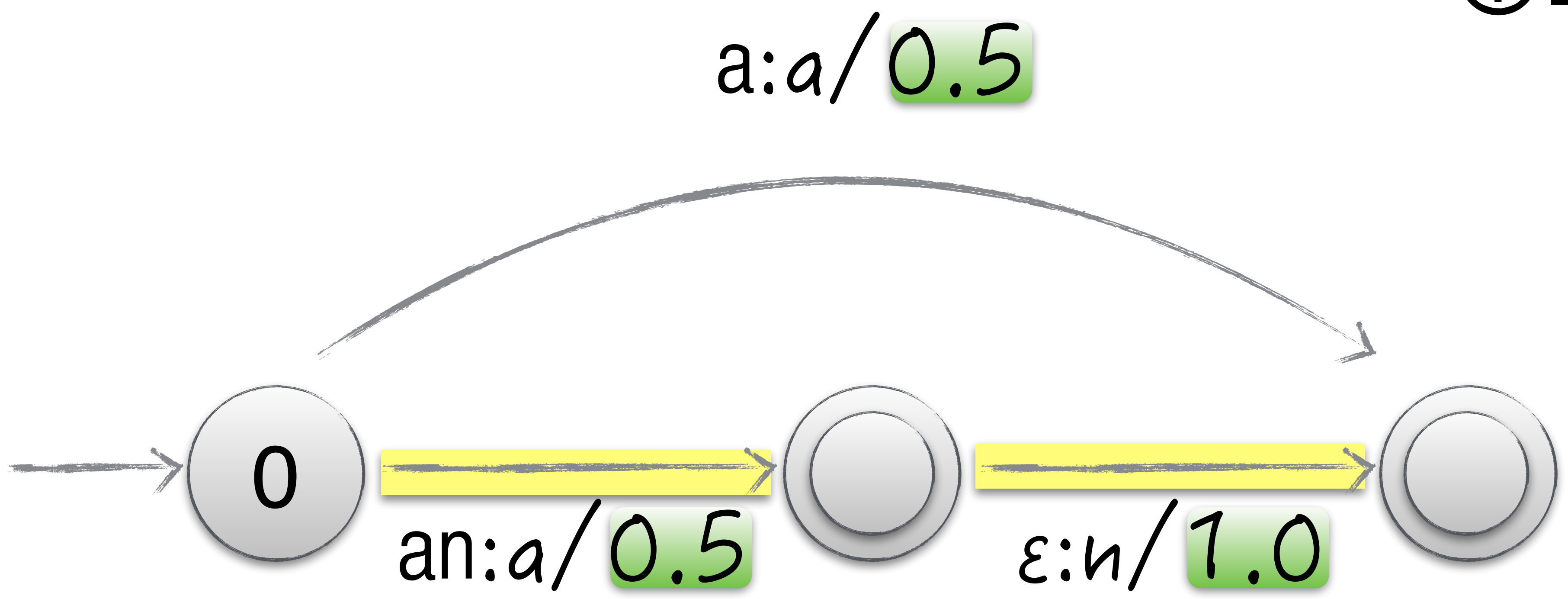
"Viterbi Approx." of -log Pr[y|x]

Operator $\oplus_{\log}$ defined as: $x \oplus_{\log} y = -\log (e^{-x} + e^{-y})$

[M02] Mohri, M. Semiring frameworks and algorithms for shortest-distance problems, Journal of Automata, Languages and Combinatorics, 7(3):321—350, 2002.

Weighted Path: Tropical Semiring

$\oplus$=min | $\otimes$=+



- Weight of a path π is the $\otimes$-product of all the transitions in π

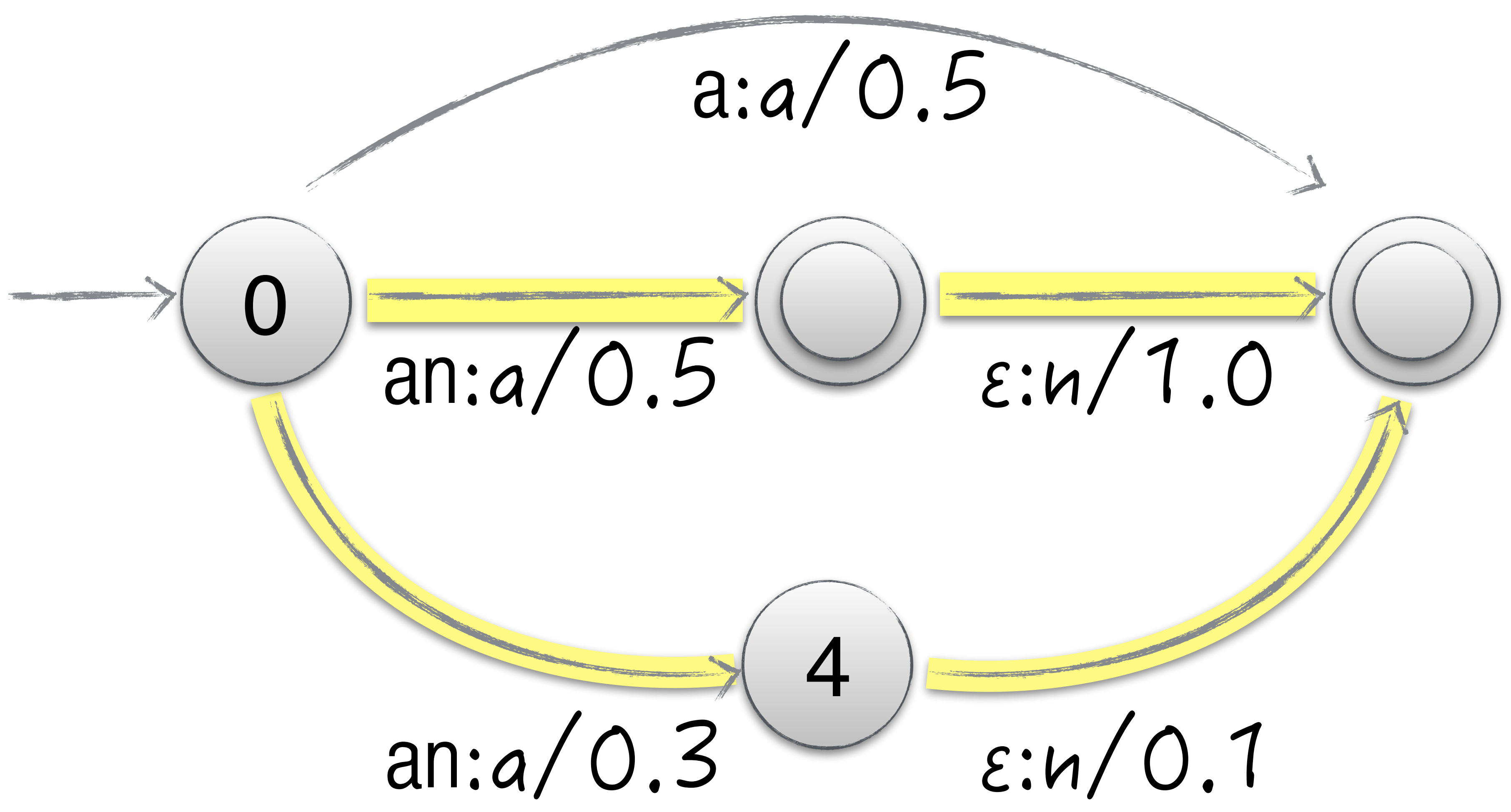
  $w(\pi)$: $(0.5 \otimes 1.0) = 0.5 + 1.0 = 1.5$

- Weight of a sequence "x,y" is the $\oplus$-sum of all paths labeled with "x,y"

  w((an), (a n)) = $(1.5 \oplus \bar{0}) = \min(1.5, \infty) = 1.5$

Weighted Path: Tropical Semiring

$\oplus$=min | $\otimes$=+



- Weight of a sequence "x,y" is the $\oplus$-sum of all paths labeled with "x,y"

  w((an), (a n)) = ?

  Path 1: (0.5 $\otimes$ 1.0) = 1.5

  Path 2: (0.3 $\otimes$ 0.1) = 0.4

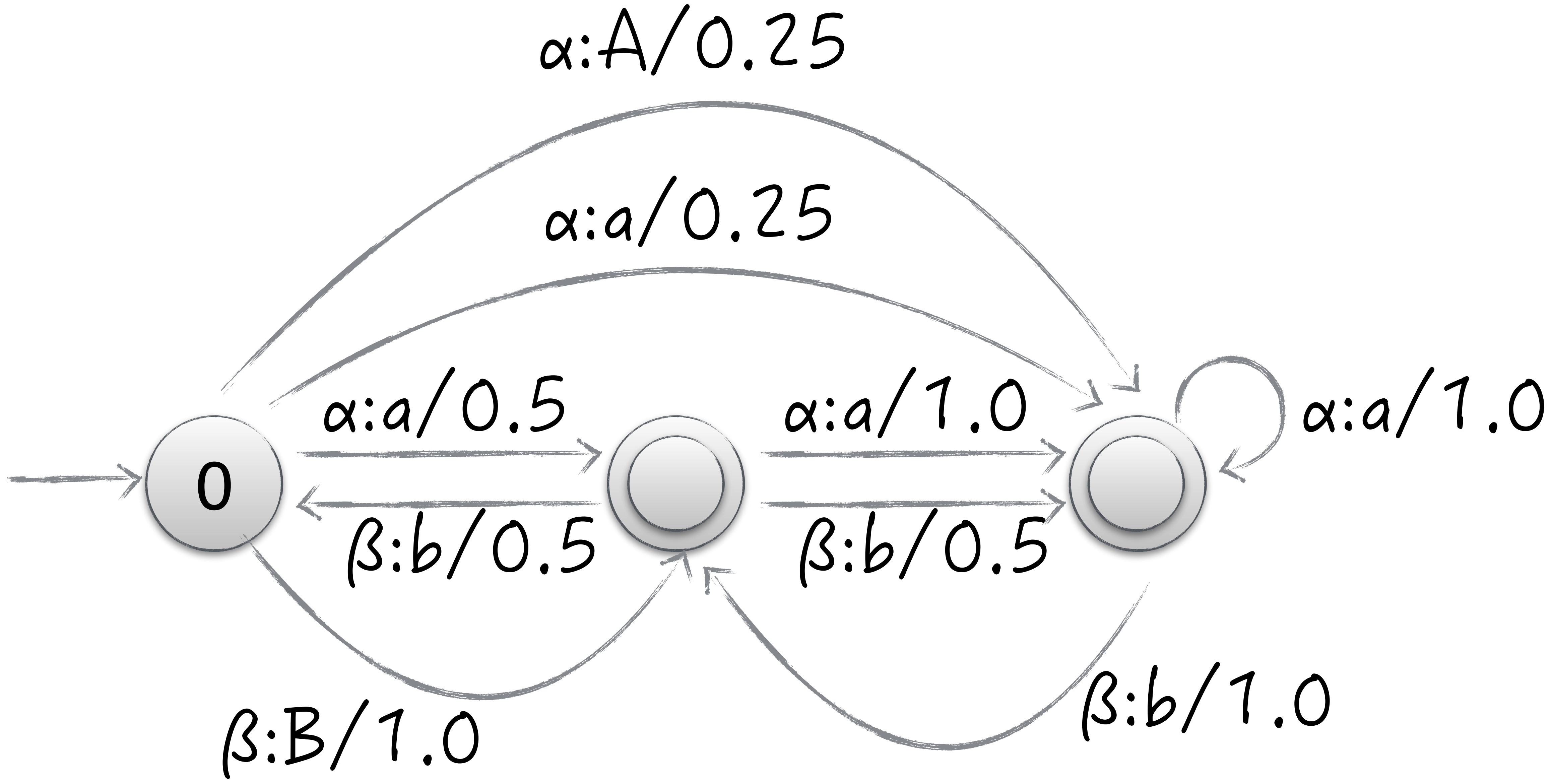
  Weight of "((an), (a n))" = (1.5 $\oplus$ 0.4) = 0.4

Shortest Path

- Recall $T(x,y) = \bigoplus_{\pi \in P(x,y)} w(\pi)$
  where $P(x,y)$ = set of paths with input/output (x,y); $w(\pi) = \bigotimes_{e \in \pi} w(e)$

- In the tropical semiring $\oplus$ is min. $T(x,y)$ associated with a single path in $P(x,y)$: Shortest Path
  - Can be found using Dijkstra's algorithm : $\Theta(|E| + |Q| \cdot \log|Q|)$ time

Shortest Path

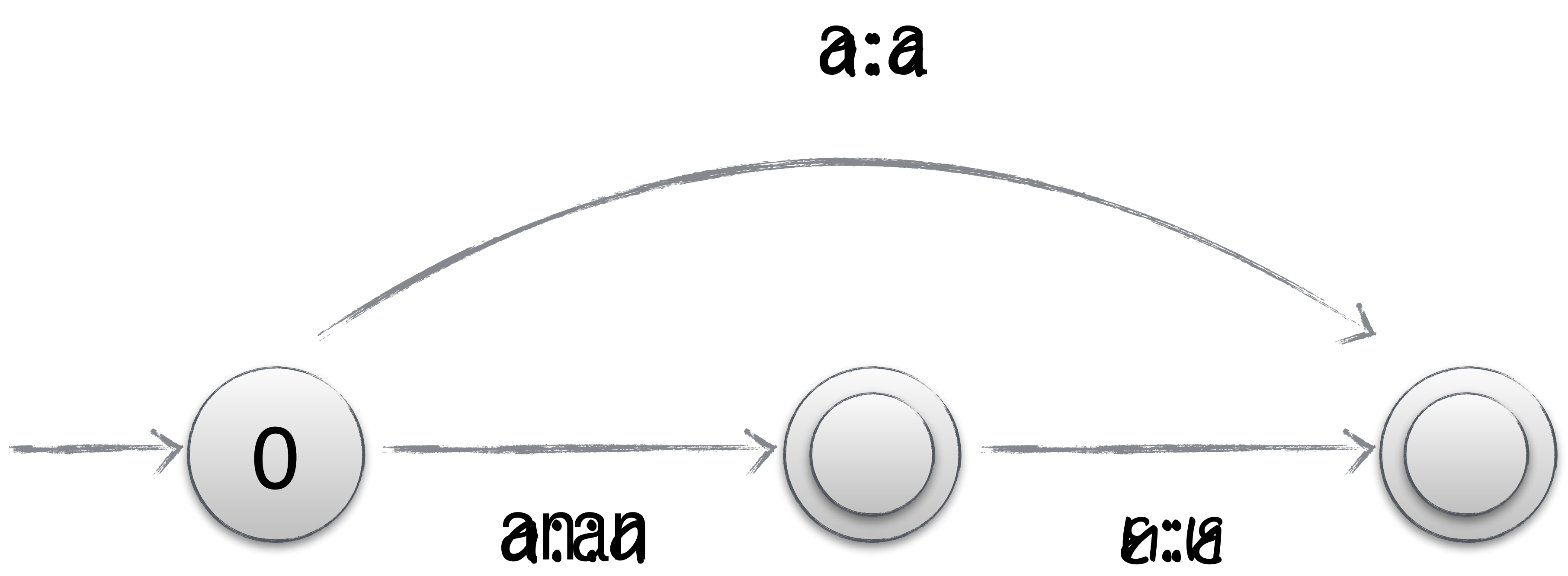


T("α", "a") = ?

T("αα", "aa") = ?

Inversion

Swap the input and output labels in each transition

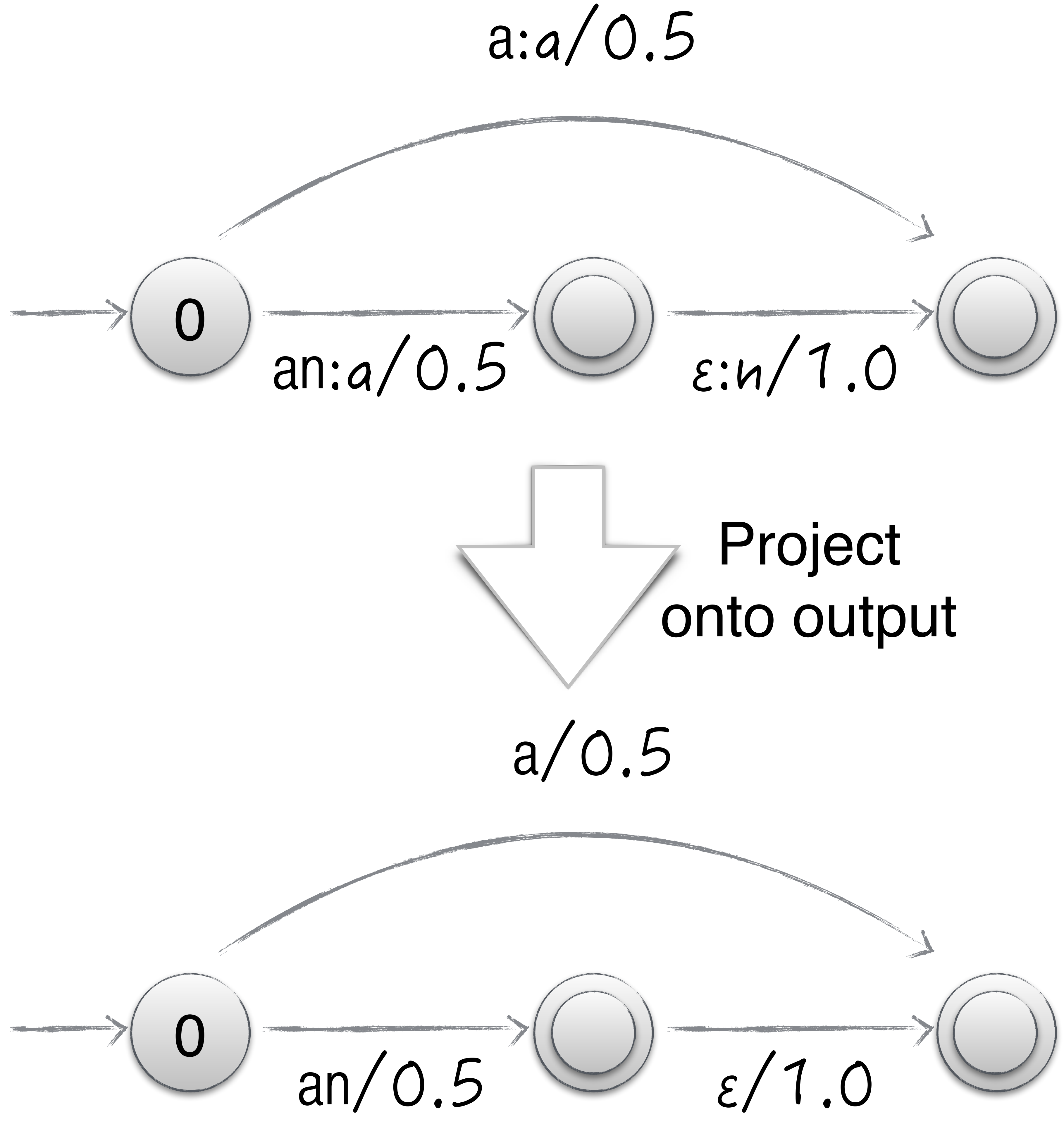


Weights (if they exist) are retained on the arcs

This operation comes in handy, especially during composition!

Projection

Project onto the input or output alphabet



Basic FST Operations (Rational Operations)

The set of weighted transducers are closed under the following operations [Mohri ‘02]:

1. Sum or Union: $(T_1 \oplus T_2)(\underline{x}, \underline{y}) = T_1(\underline{x}, \underline{y}) \oplus T_2(\underline{x}, \underline{y})$

2. Product or Concatenation: $(T_1 \otimes T_2)(x, y) = \bigoplus_{\substack{x=x_1x_2 \\ y=y_1y_2}} T_1(x_1, y_1) \otimes T_2(x_2, y_2)$

3. Kleene-closure: $T^*(x, y) = \bigoplus_{n=0}^{\infty} T^n(x, y)$

Basic FST Operations (Rational Operations)

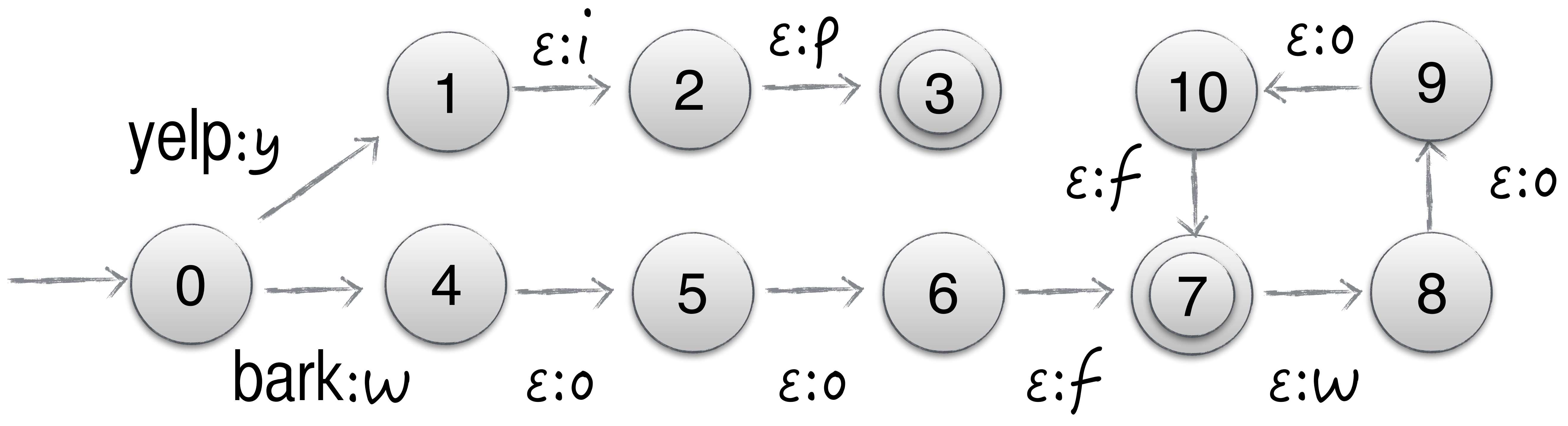
The set of weighted transducers are closed under the following operations [Mohri '02]:

1. Sum or Union: $(T_1 \oplus T_2)(\underline{x}, \underline{y}) = T_1(\underline{x}, \underline{y}) \oplus T_2(\underline{x}, \underline{y})$

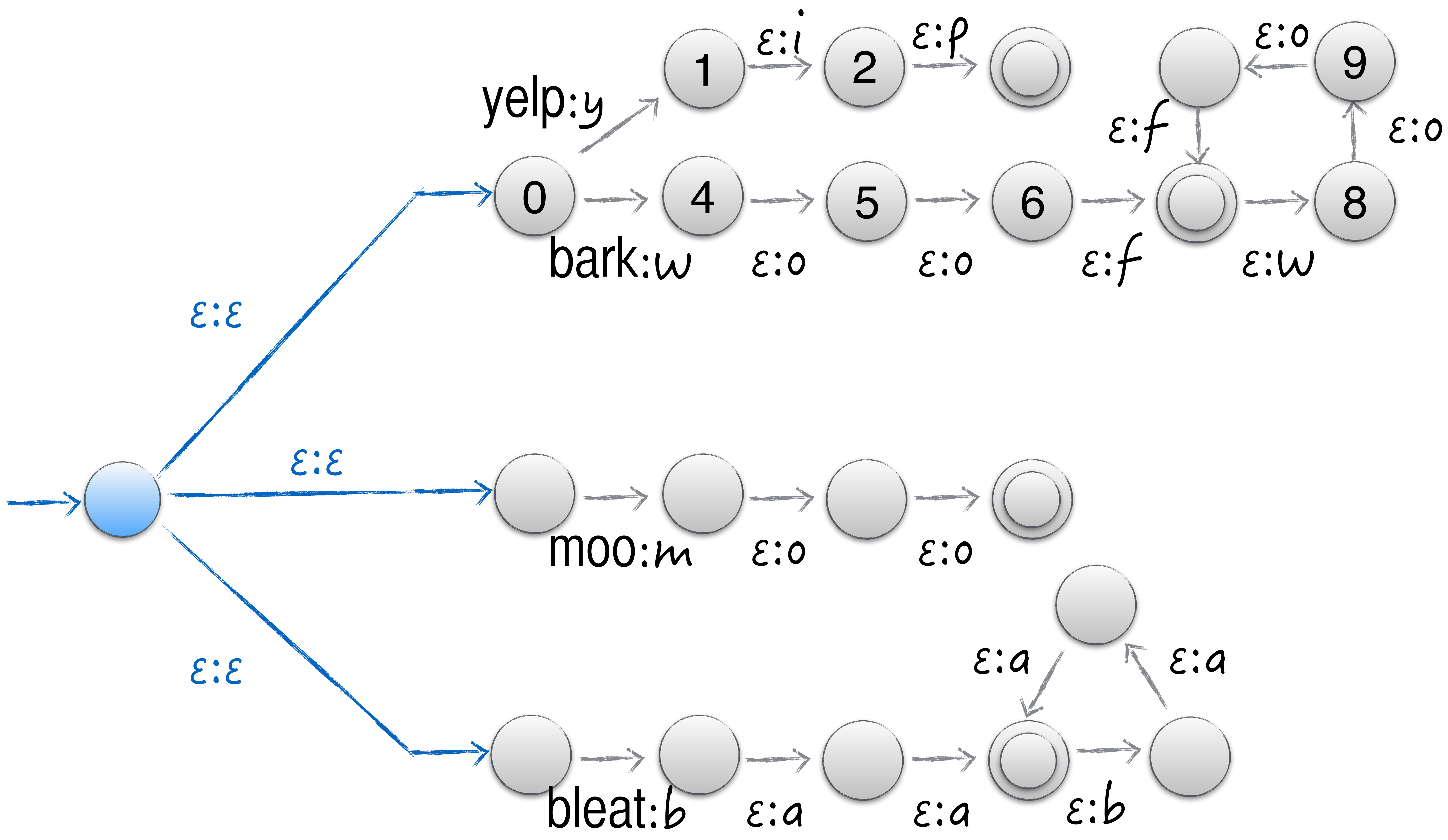
2. Product or Concatenation: $(T_1 \otimes T_2)(x, y) = \bigoplus_{\substack{x=x_1x_2 \\ y=y_1y_2}} T_1(x_1, y_1) \otimes T_2(x_2, y_2)$

3. Kleene-closure: $T^*(x, y) = \bigoplus_{n=0}^{\infty} T^n(x, y)$

Example: Recall Barking Dog



Example: Union



Animal farm!

Basic FST Operations (Rational Operations)

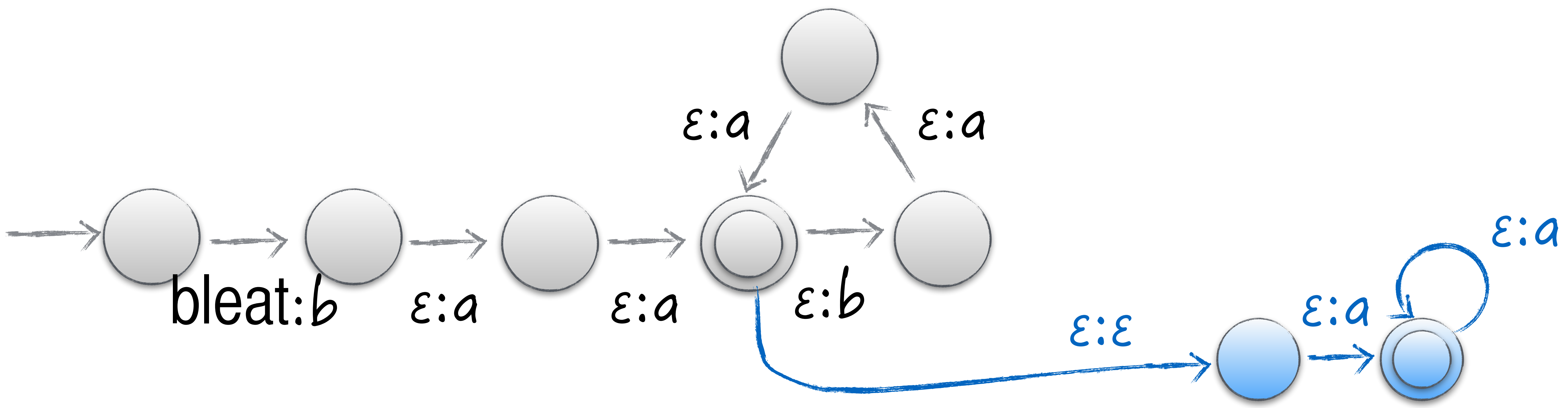
The set of weighted transducers are closed under the following operations [Mohri '02]:

1. Sum or Union: $(T_1 \oplus T_2)(\underline{x}, \underline{y}) = T_1(\underline{x}, \underline{y}) \oplus T_2(\underline{x}, \underline{y})$

2. Product or Concatenation: $(T_1 \otimes T_2)(x, y) = \bigoplus_{\substack{x=x_1x_2 \\ y=y_1y_2}} T_1(x_1, y_1) \otimes T_2(x_2, y_2)$

3. Kleene-closure: $T^*(x, y) = \bigoplus_{n=0}^{\infty} T^n(x, y)$

Example: Concatenation



Suppose the last "*baa*" in a bleat should be followed by one or more a's

(e.g., "*baabaa*" is not OK, but "*baaa*" and "*baabaaaaa*" are)

Basic FST Operations (Rational Operations)

The set of weighted transducers are closed under the following operations [Mohri ‘02]:

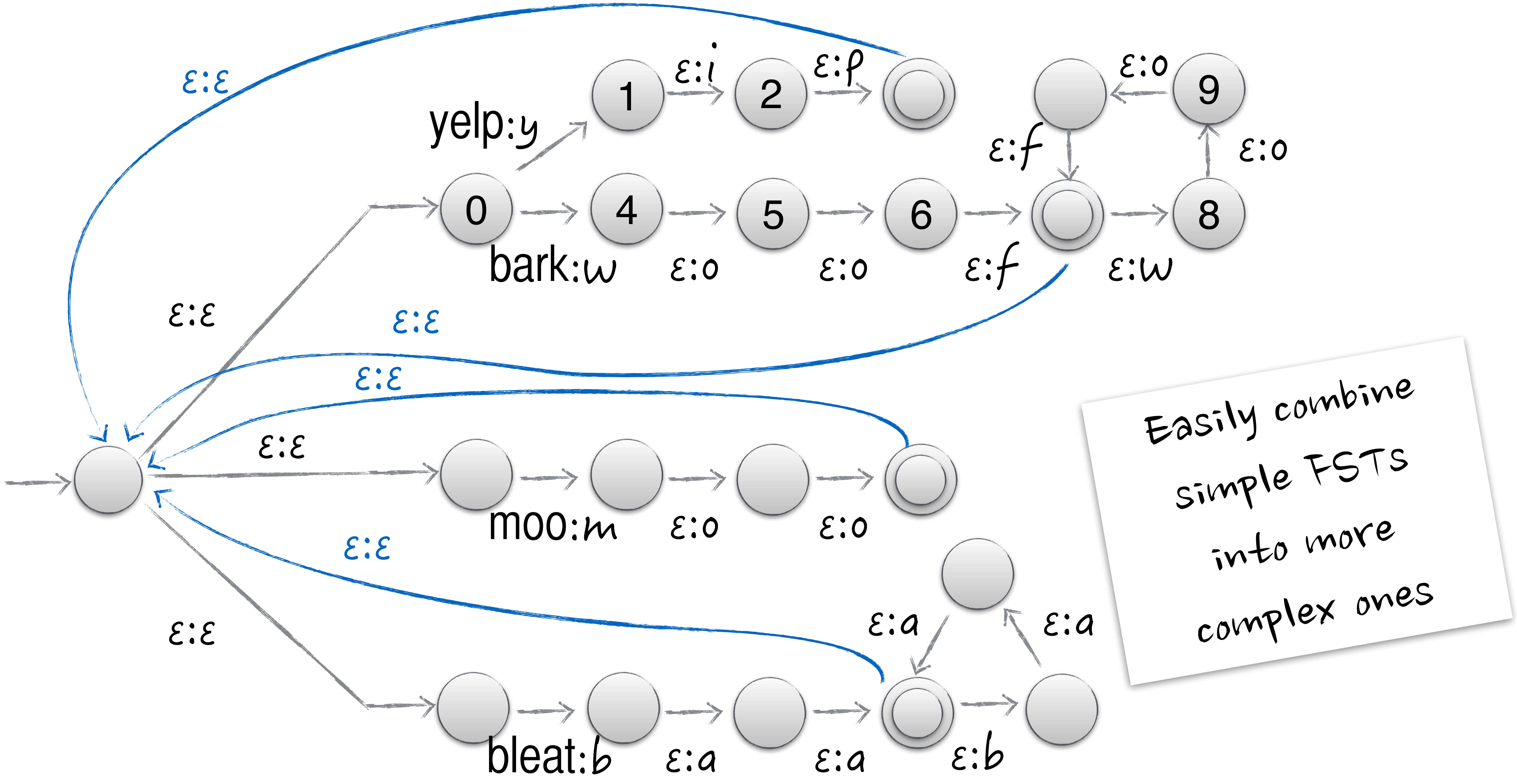
1. Sum or Union: $(T_1 \oplus T_2)(\underline{x}, \underline{y}) = T_1(\underline{x}, \underline{y}) \oplus T_2(\underline{x}, \underline{y})$

2. Product or Concatenation: $(T_1 \otimes T_2)(x, y) = \bigoplus_{\substack{x=x_1x_2 \\ y=y_1y_2}} T_1(x_1, y_1) \otimes T_2(x_2, y_2)$

3. Kleene-closure: $T^*(x, y) = \bigoplus_{n=0}^{\infty} T^n(x, y)$

$x=x_1x_2$
$y=y_1y_2$

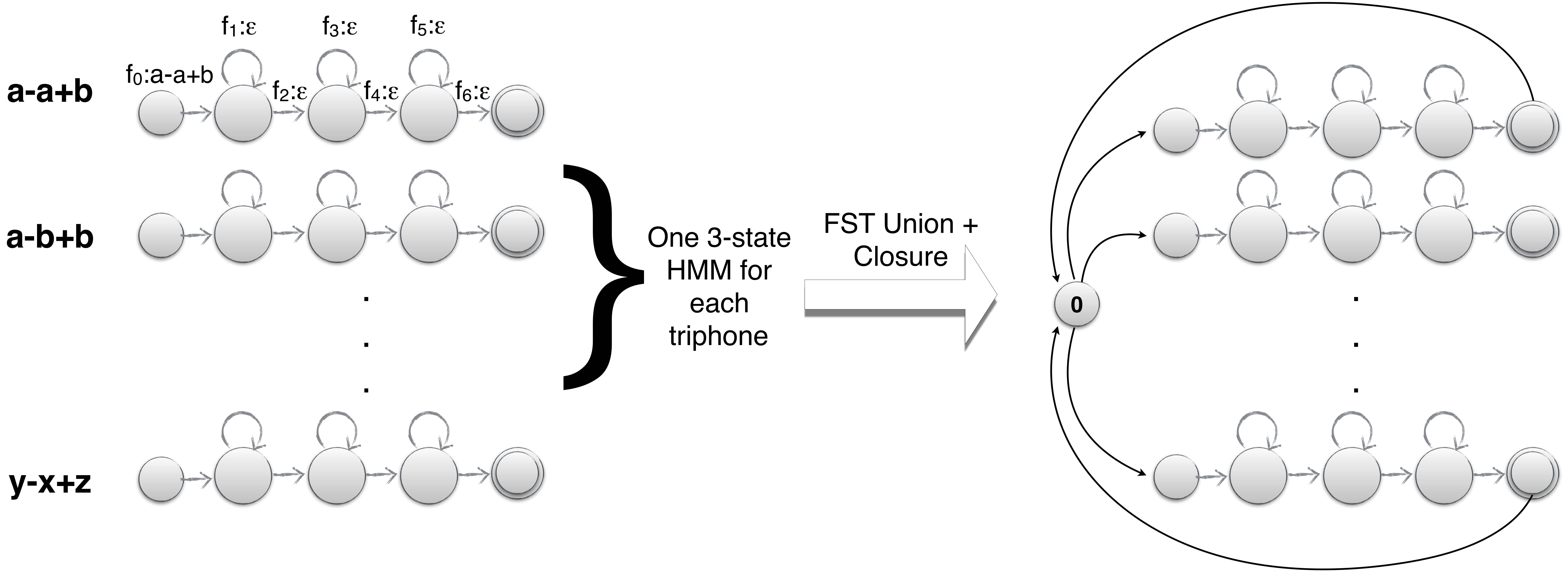
Example: Closure



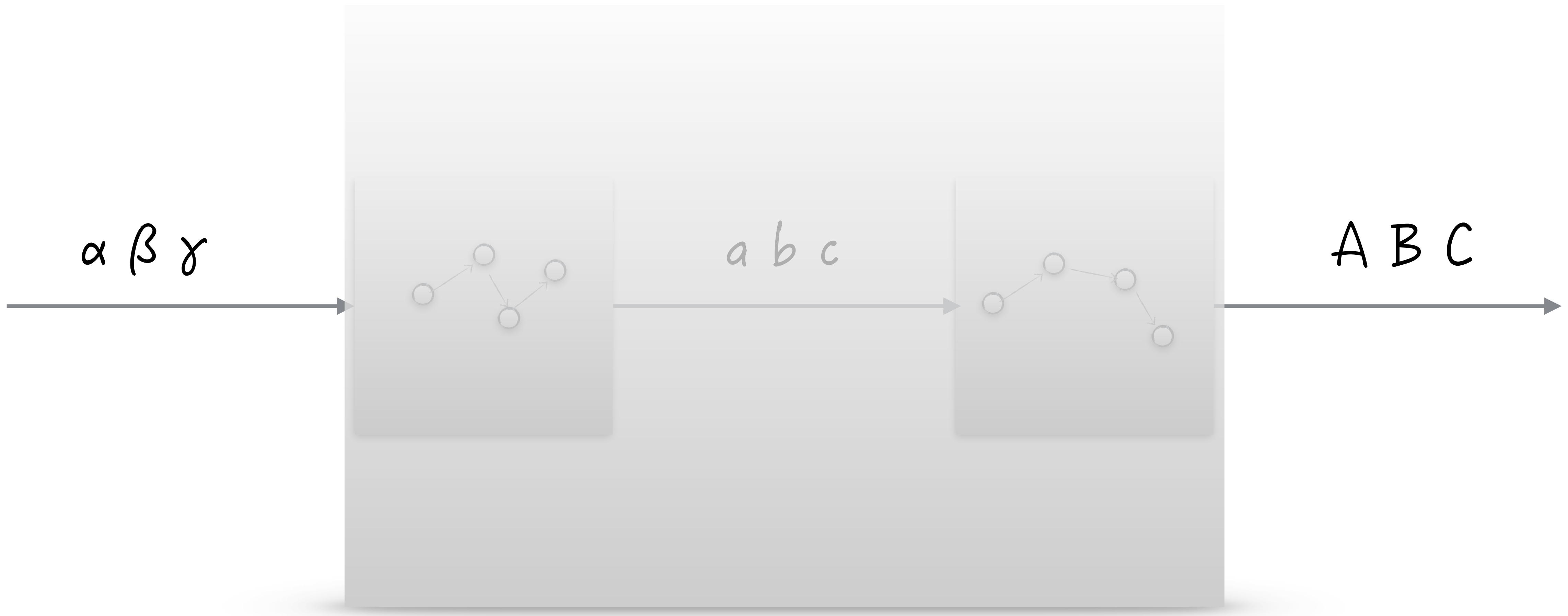
Animal farm: allow arbitrarily long sequence of sounds!

bark moo yelp bleat → w o o f w o o f m o o y i p b a a b a a

Acoustic Model WFST



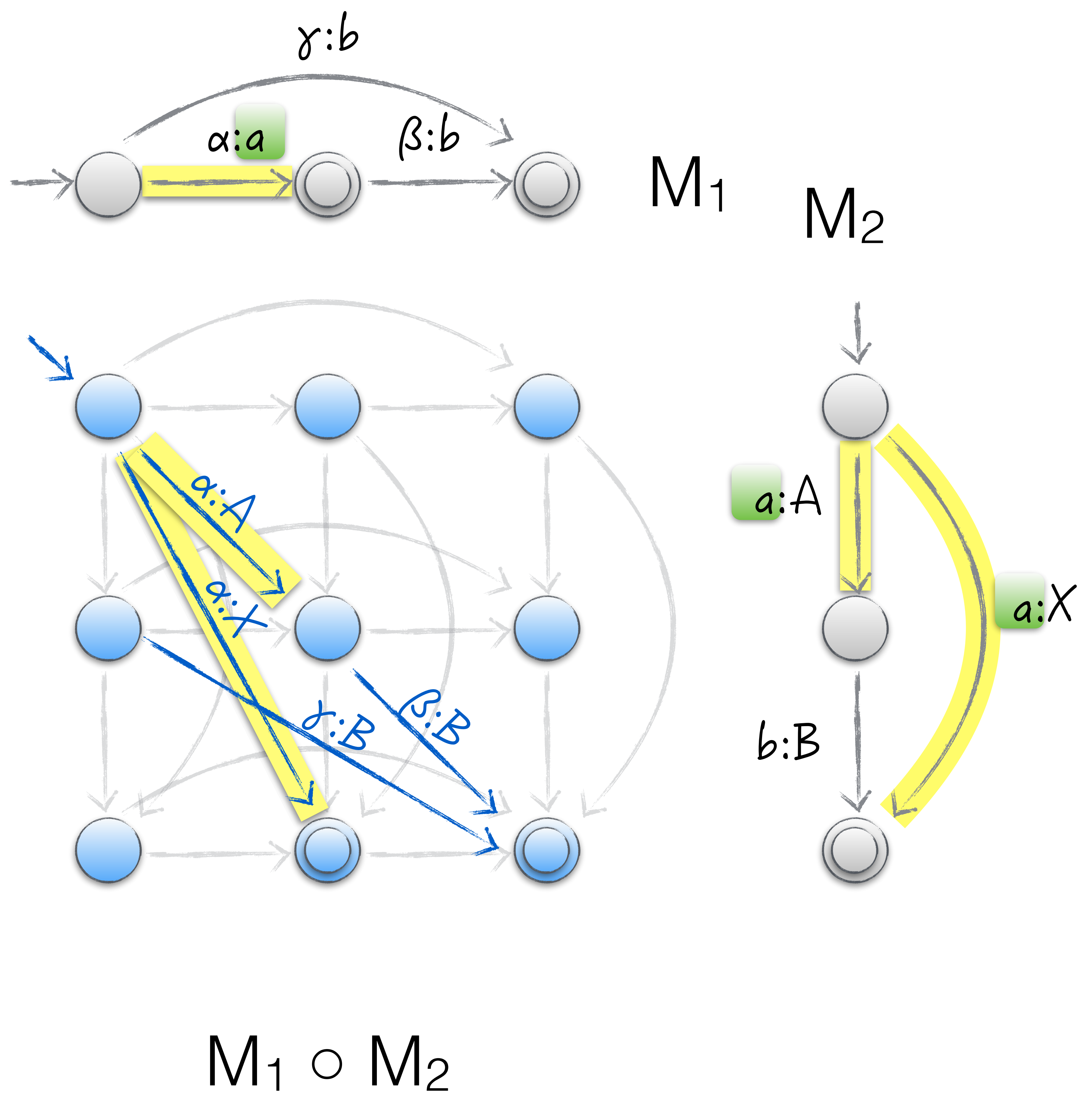
Composition



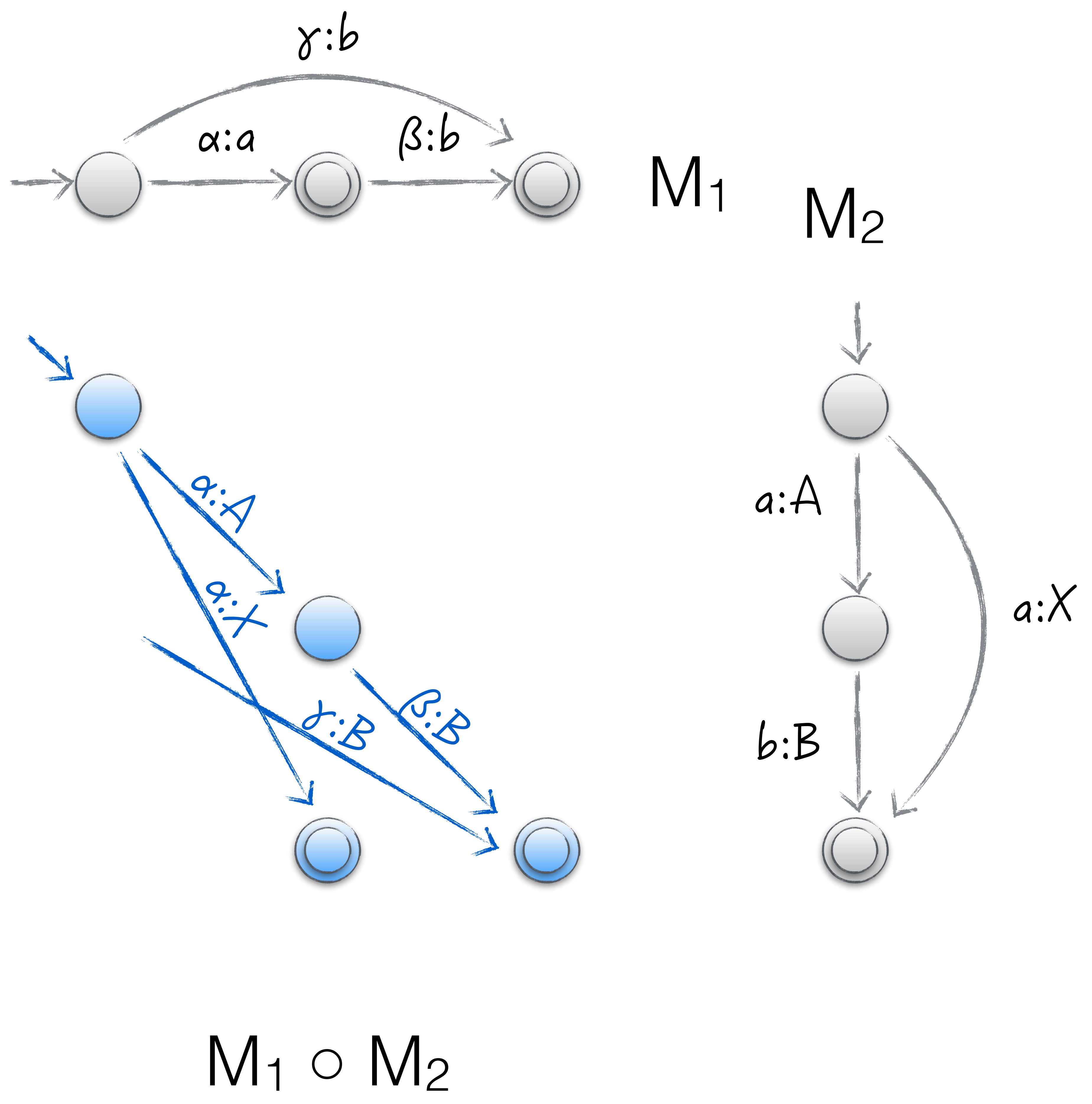
- If T_1 transduces x to z, and T_2 transduces z to y, then $T_1 \circ T_2$ transduces x to y

- $(T_1 \circ T_2)(x, y) = \bigoplus_{z} T_1(x, z) \otimes T_2(z, y)$

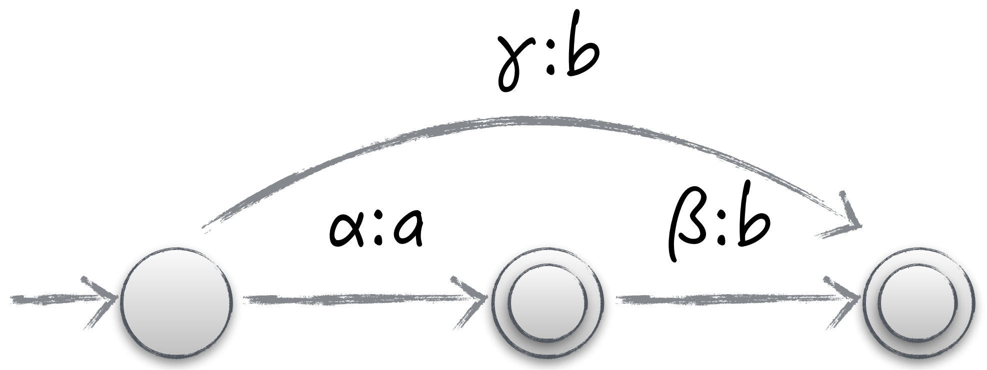
Composition: Construction



Composition



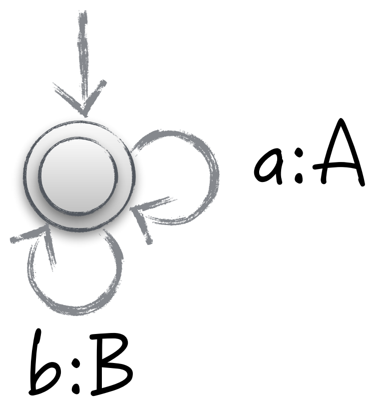
Composition: Example 1

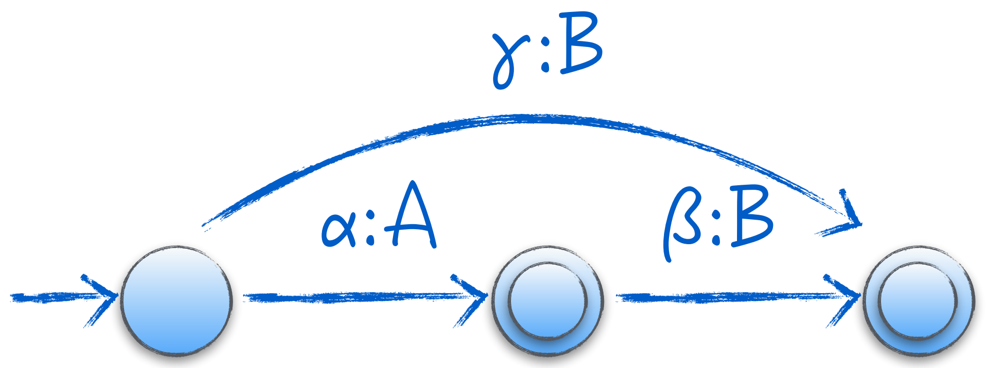


M_1

Tz translates the output from one alphabet to another

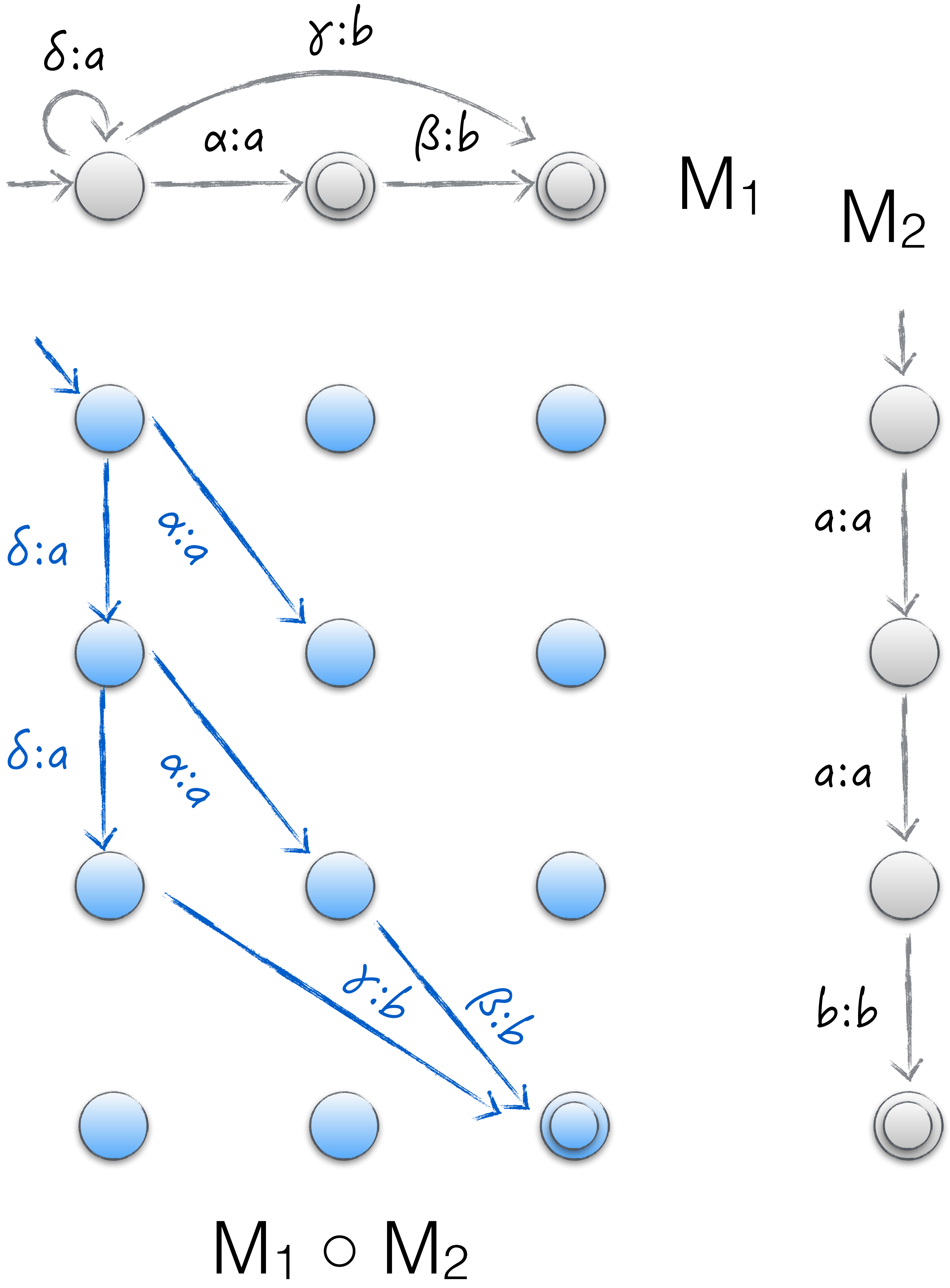
M_2





$M_1 \circ M_2$

Composition: Example 2



T2 restricts the output string to aab.
T1 ○ T2 is a "lattice".