WFSTs for ASR

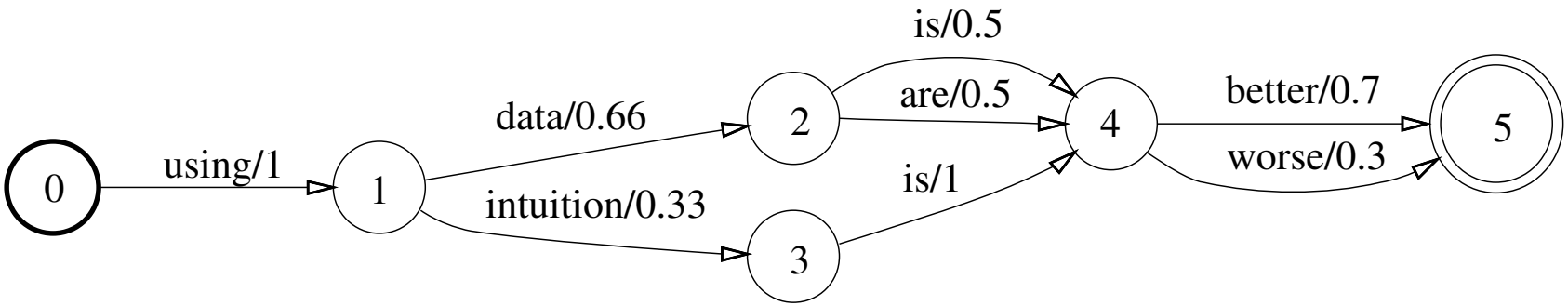
Peter Bell

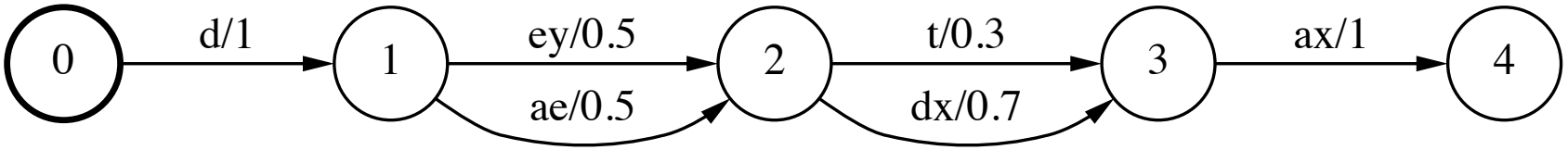
Automatic Speech Recognition – ASR Lecture 9
12 February 2024

Weighted Finite State Transducers

- 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
- There is a single start state. Any state can optionally be a final state (with a weight)
- Used by Kaldi

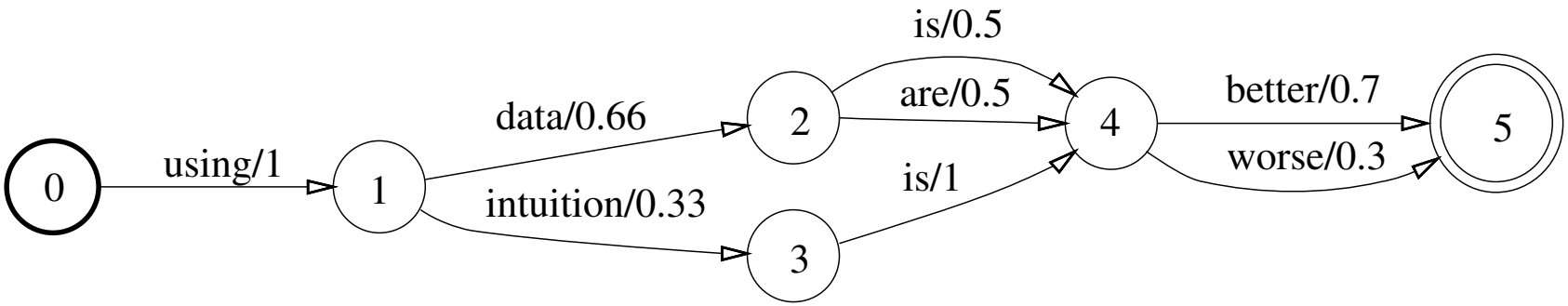
Weighted Finite State Acceptors



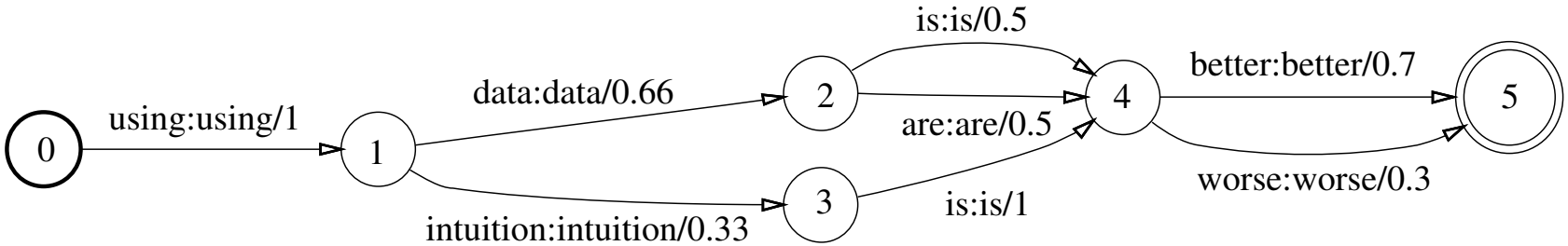


Weighted Finite State Transducers

Acceptor

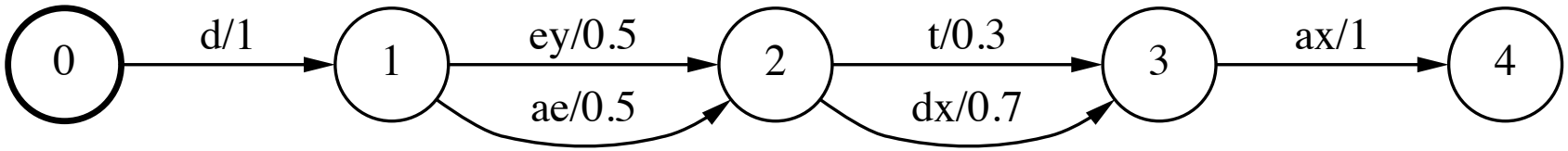


Transducer

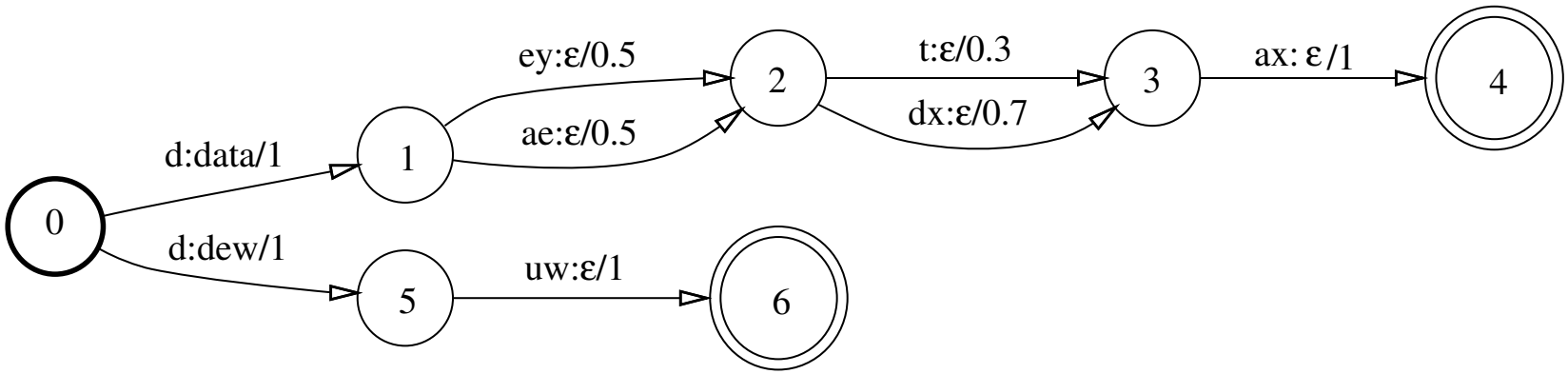


Weighted Finite State Transducers

Acceptor



Transducer



FST Weights

- Formally, WFST weights must be members of a *semiring*
- This defines special operations for mutiplication ("Times", $\otimes$) and addition ("Plus", $\oplus$)
- You can think of the weights as negative log-probabilities, so that:

$$w_1 \otimes w_2 = w_1 + w_2$$
$$w_1 \oplus w_2 = -\log(e^{-w_1} + e^{-w_2})$$

  corresponding to the normal multiplication/addition operations in the probability domain. This is the *log semiring*
- You may also encounter the *tropical semiring* (the default in OpenFst), which is the same as above, except

$$w_1 \oplus w_2 = min(w_1, w_2)$$

  which can be interpreted as taking the best of two probabilities, rather than summing them.

WFST Algorithms

Composition Combine transducers T_1 and T_2 into a single transducer acting as if the output of T_1 was passed into T_2.

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

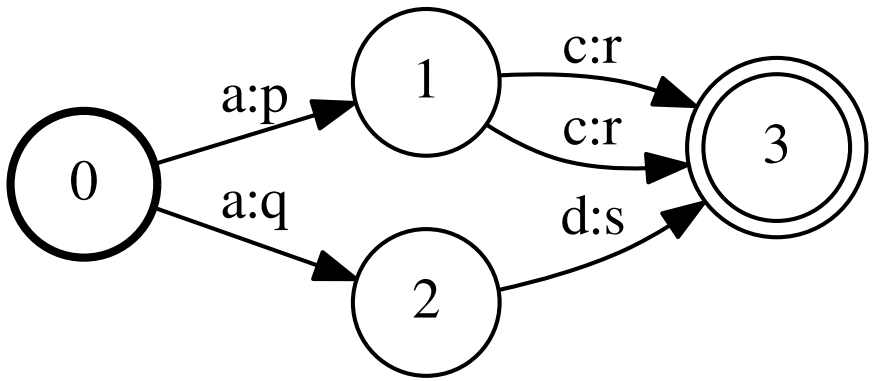
Minimisation Transforms a transducer to an equivalent transducer with the fewest possible states and transitions

Weight pushing Push the weights towards the front of the path

WFST Algorithms

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

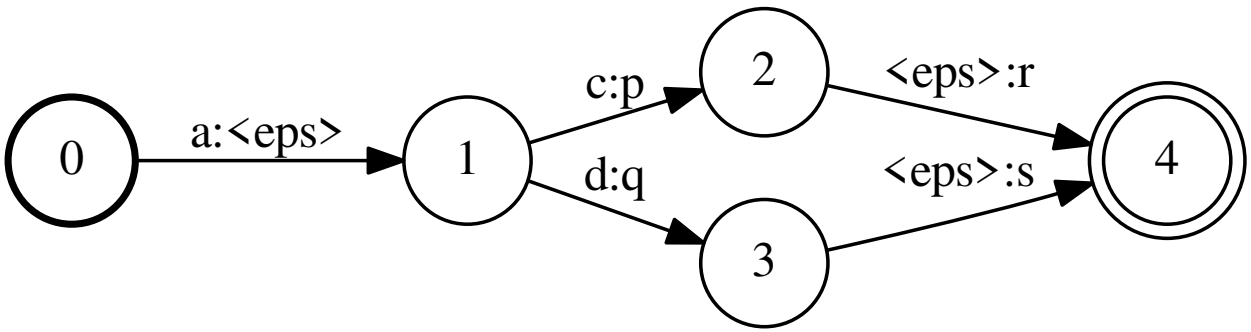
Original transducer:



WFST Algorithms

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

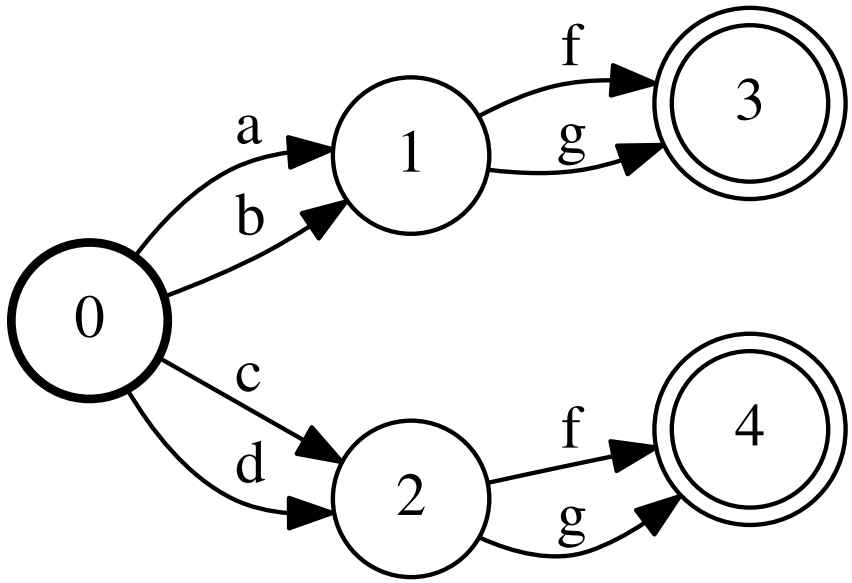
Determinized version:



WFST Algorithms

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

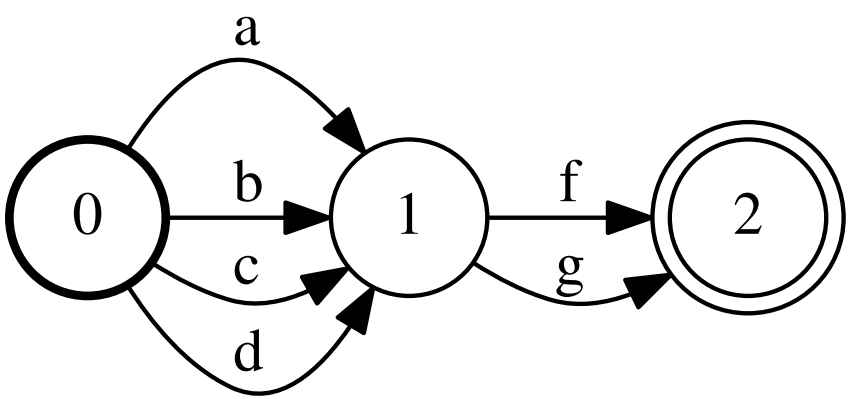
Original transducer:



WFST Algorithms

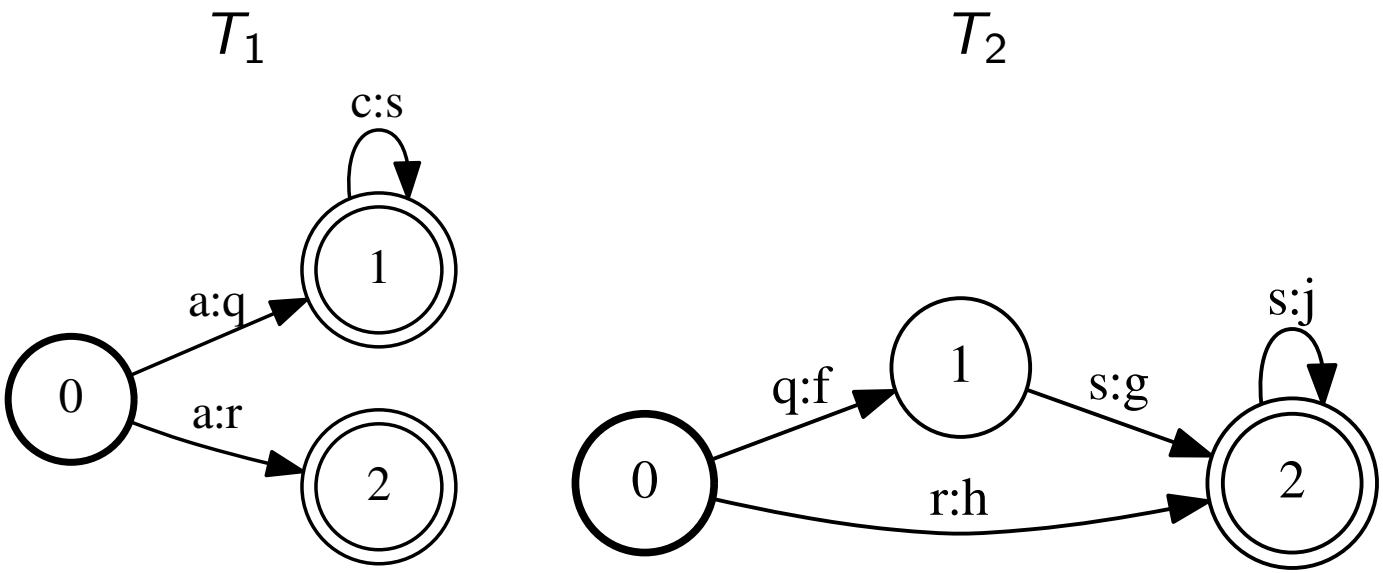
Minimisation Transforms a transducer to an equivalent transducer with the fewest possible states and transitions

Miniminized version:



WFST Algorithms

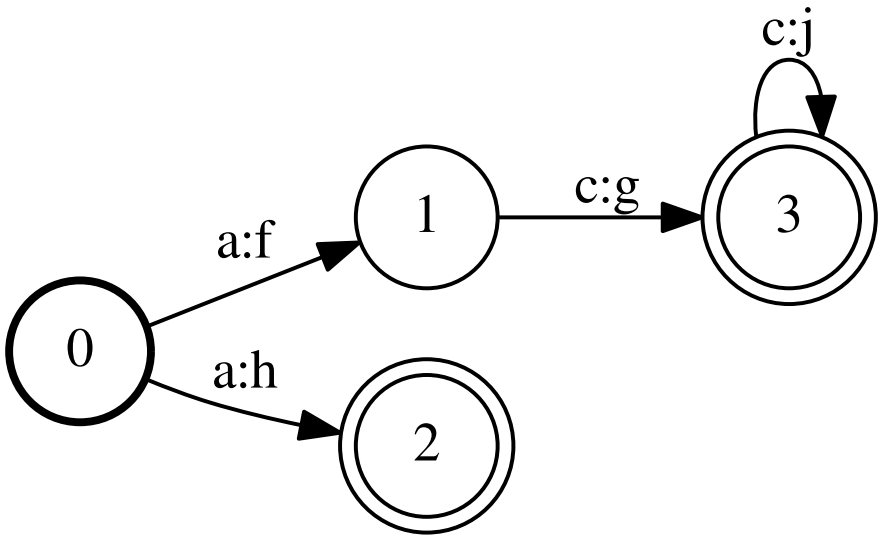
Composition Combine transducers T_1 and T_2 into a single transducer acting as if the output of T_1 was passed into T_2.



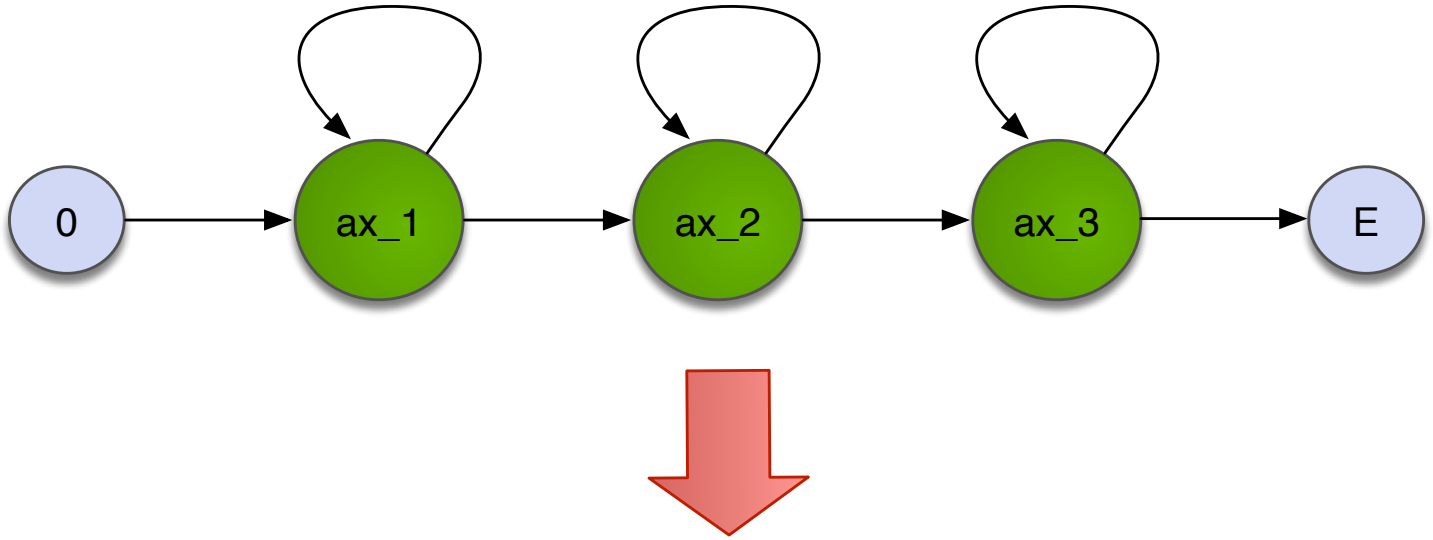
WFST Algorithms

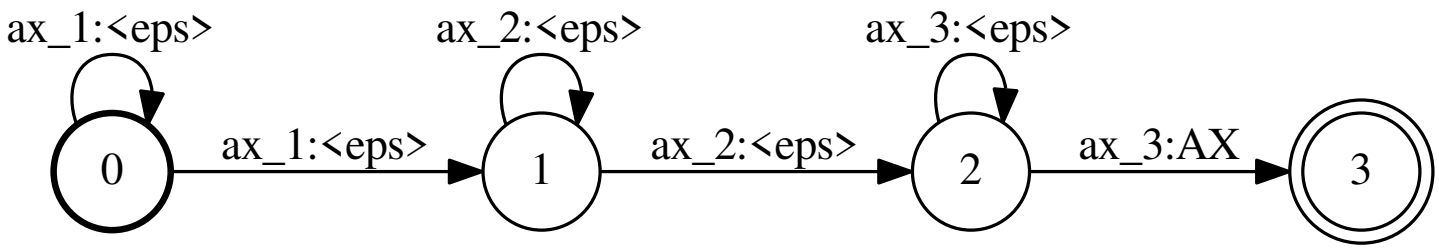
Composition Combine transducers T_1 and T_2 into a single transducer acting as if the output of T_1 was passed into T_2.

$$T_1 \circ T_2$$



The HMM as a WFST





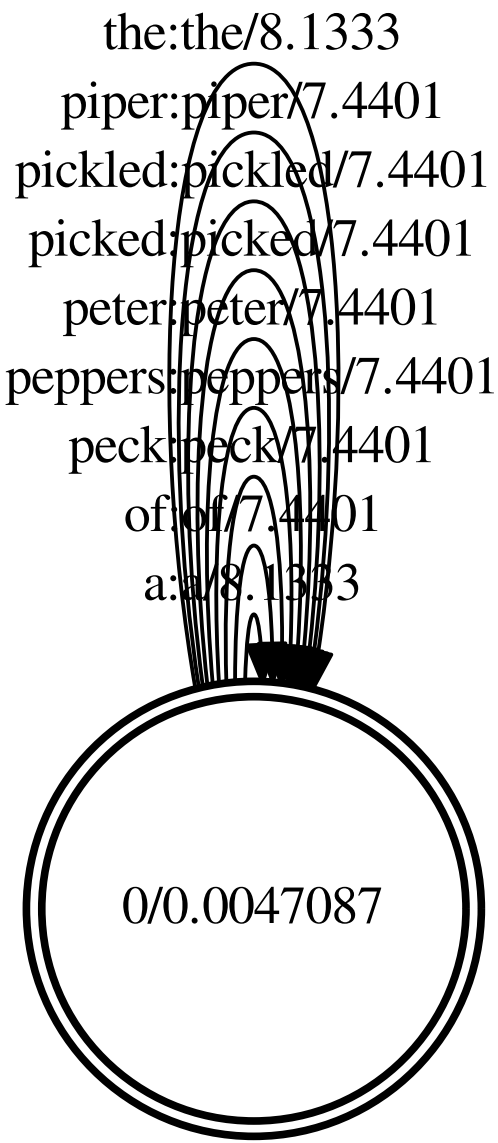
Applying WFSTs to speech recognition

- Represent the following components as WFSTs

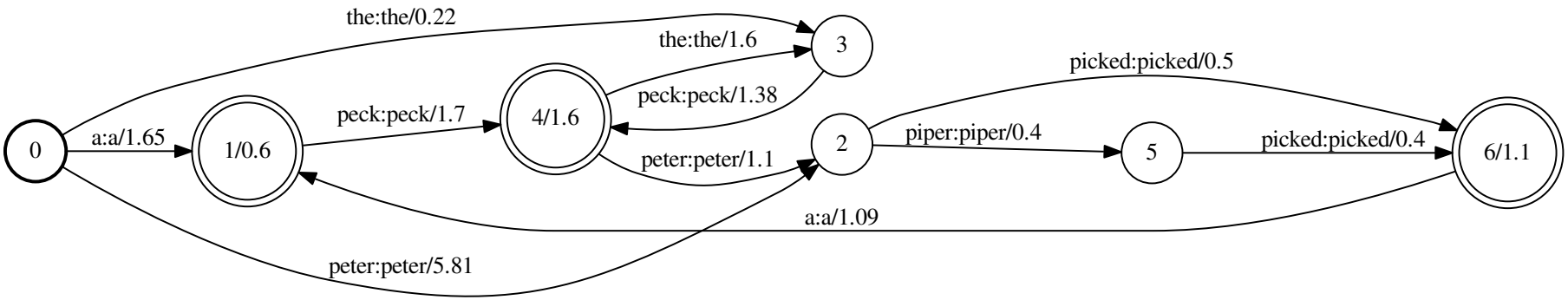
	transducer	input sequence	output sequence
G	word-level grammar	words	words
L	pronunciation lexicon	phones	words
C	context-dependency	CD phones	phones
H	HMM	HMM states	CD phones

- 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

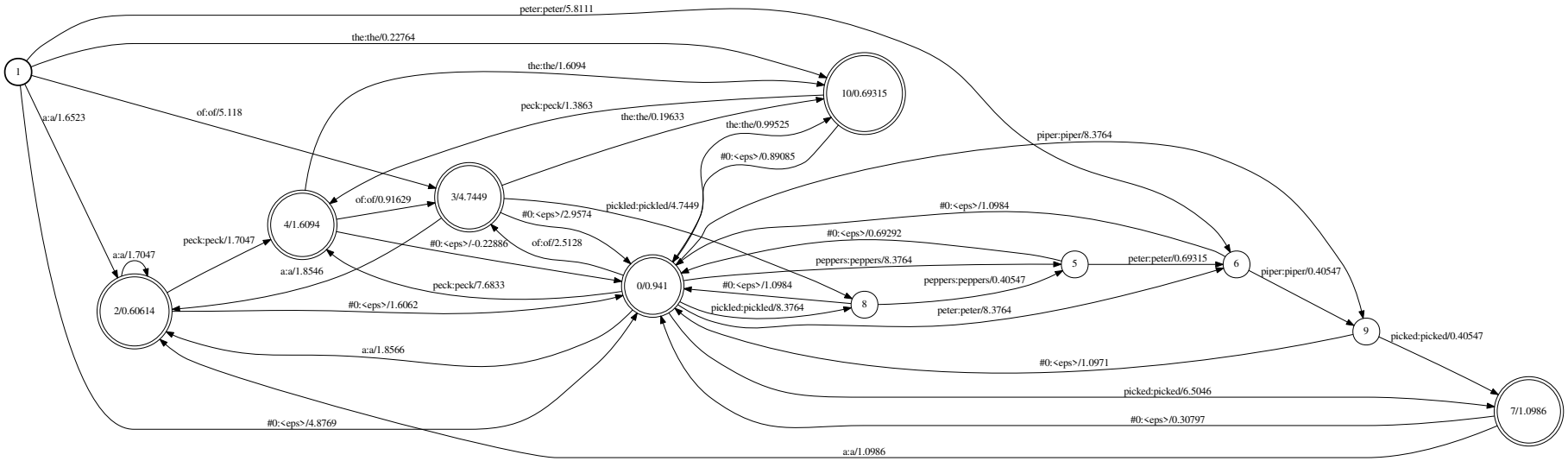
Grammar - unigram



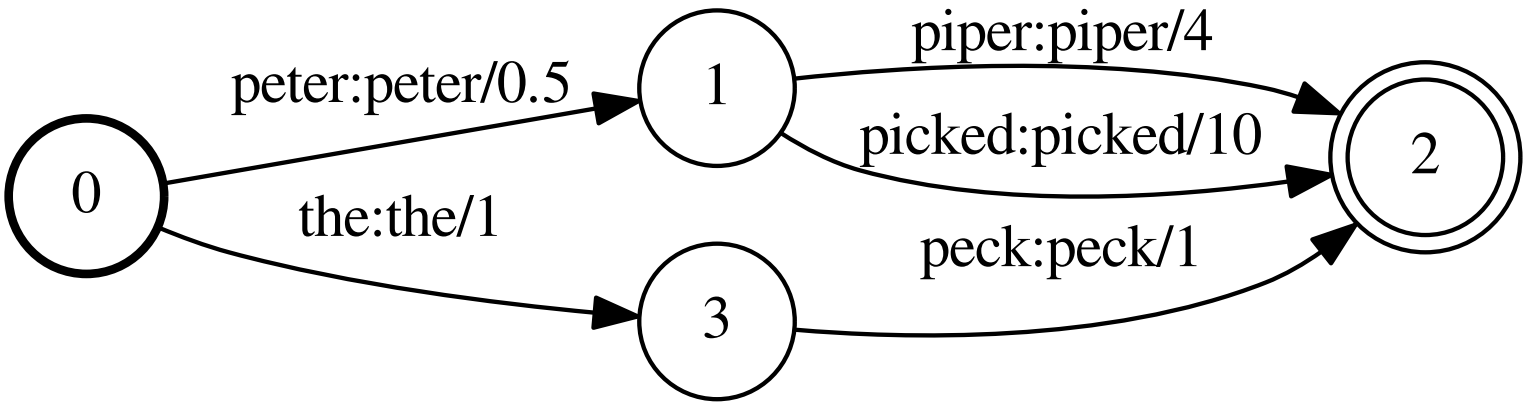
Grammar - bigram



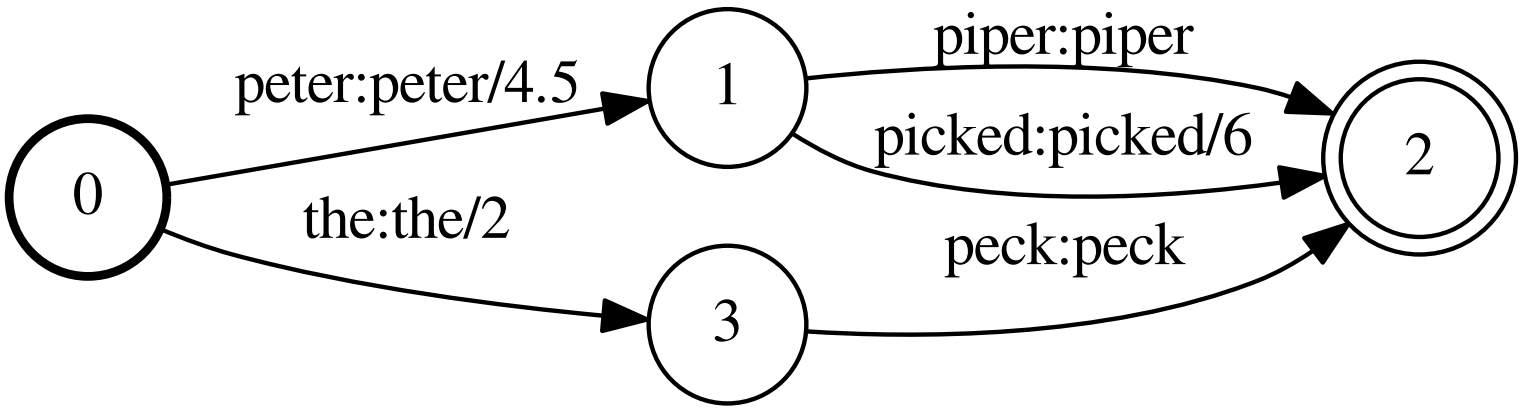
Bigram with back-off



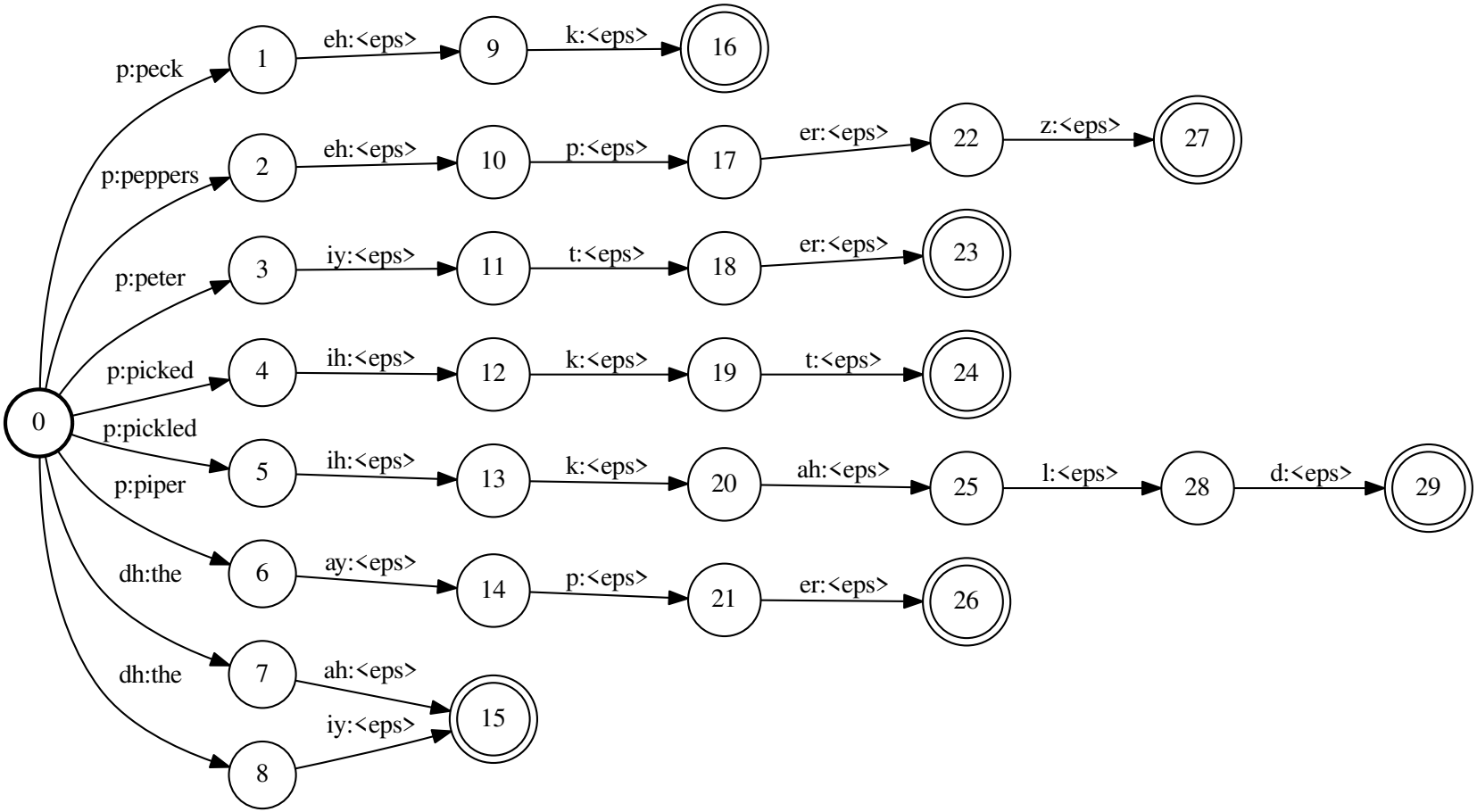
A toy example



Weight-pushed version

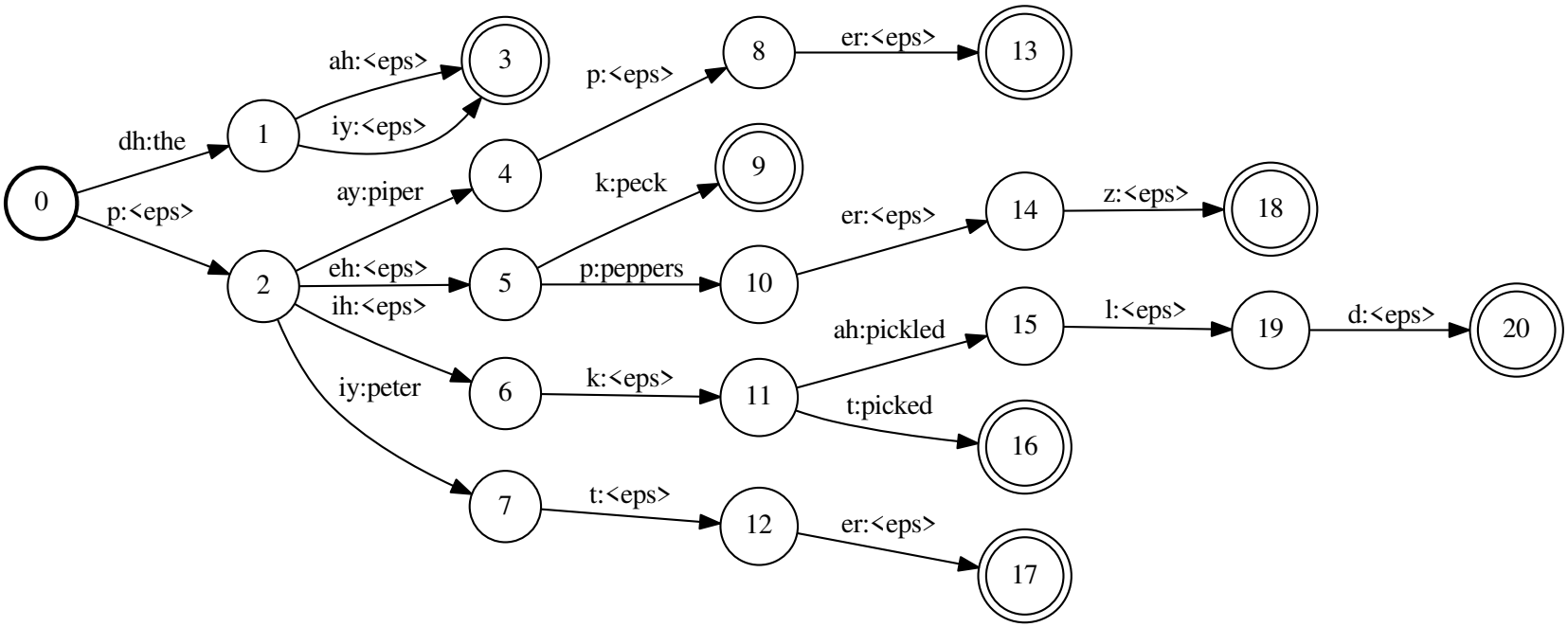


Lexicon, L



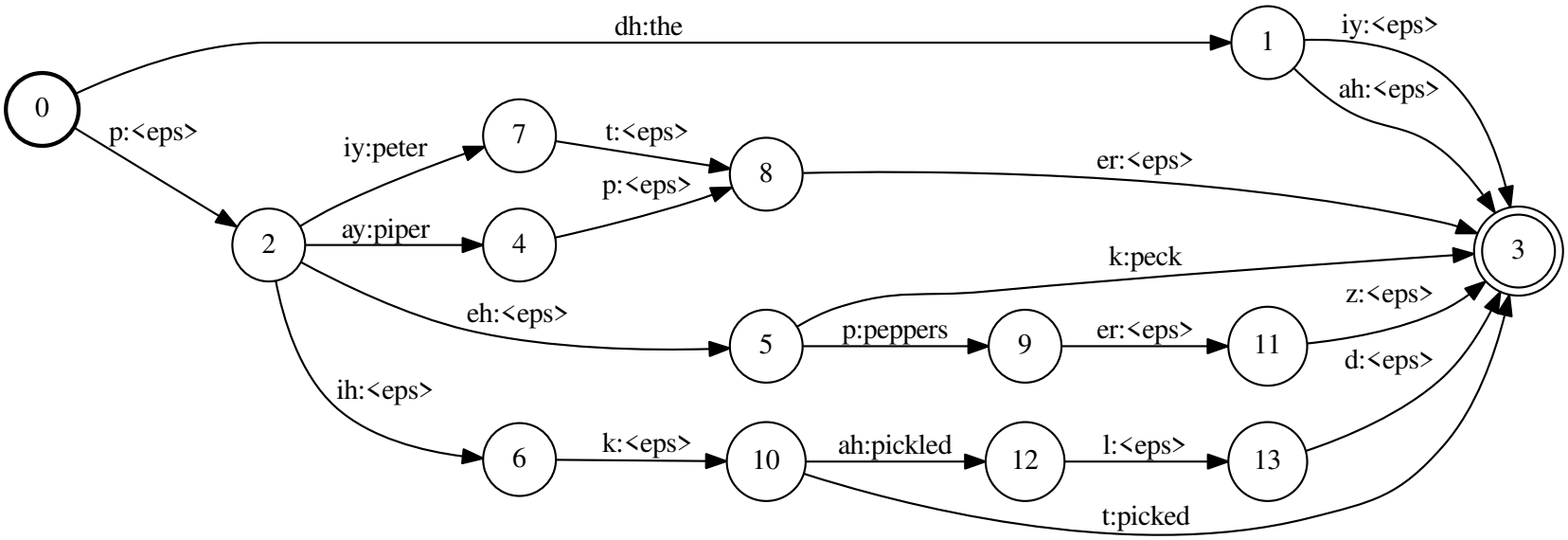
For clarity, this figure omits loops back to the start state

Determinization – $det(L)$



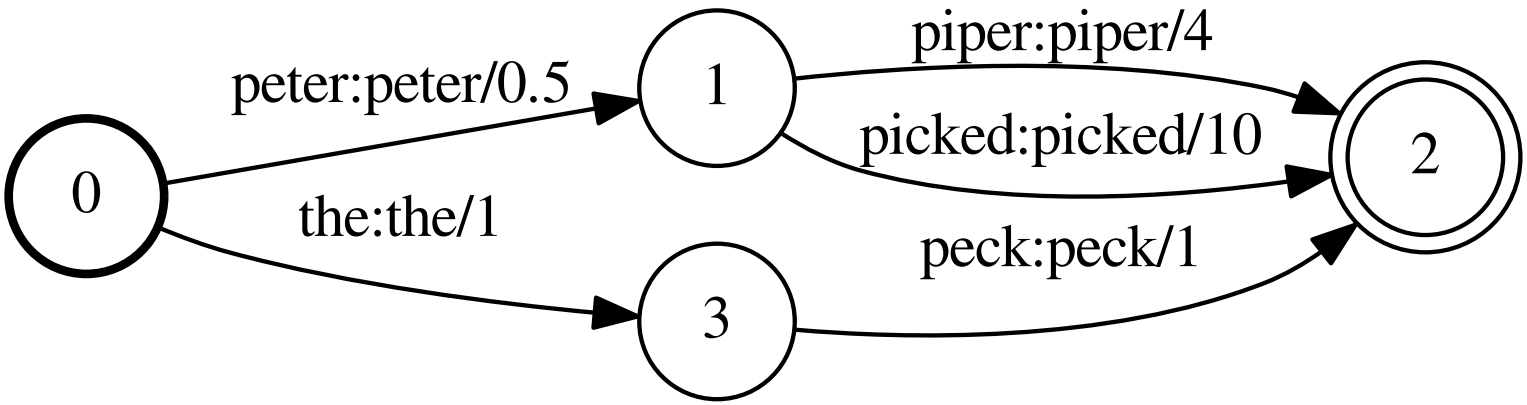
For clarity, this figure omits loops back to the start state

Minimization – $min(det(L))$

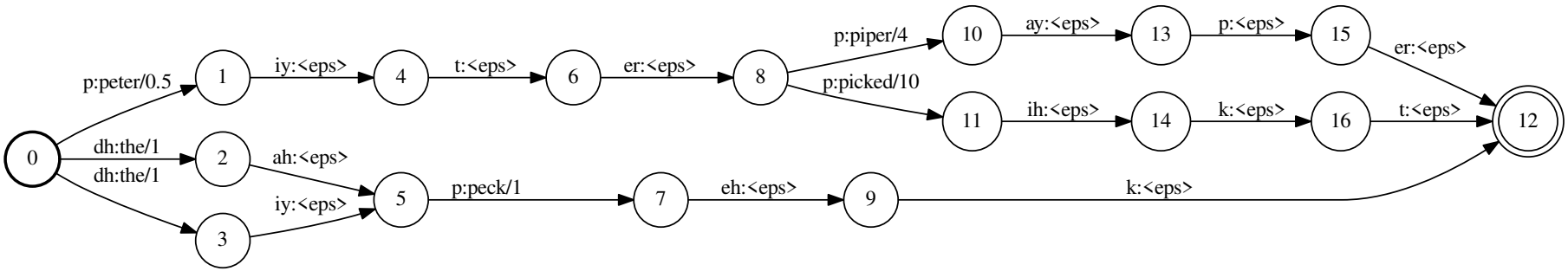


For clarity, this figure omits loops back to the start state

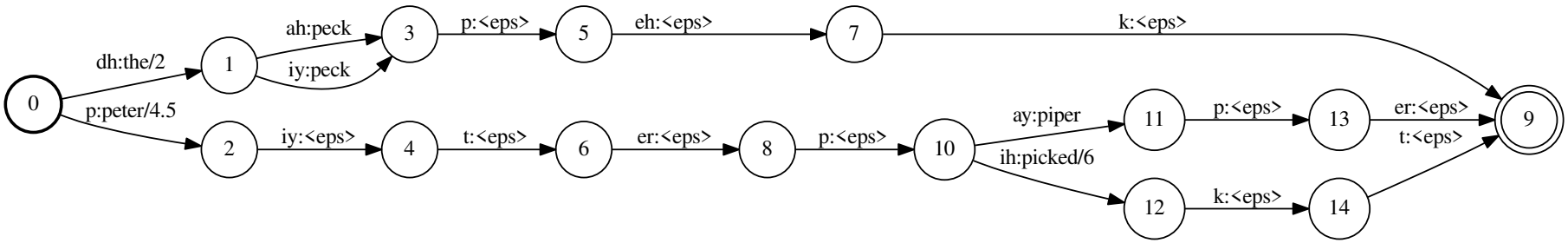
Composition



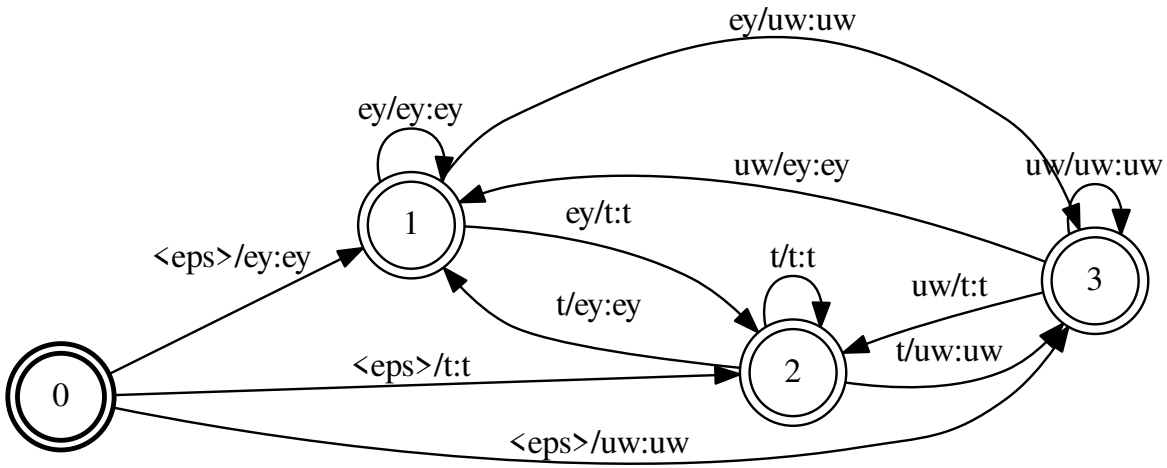
Composition: $L \circ G$



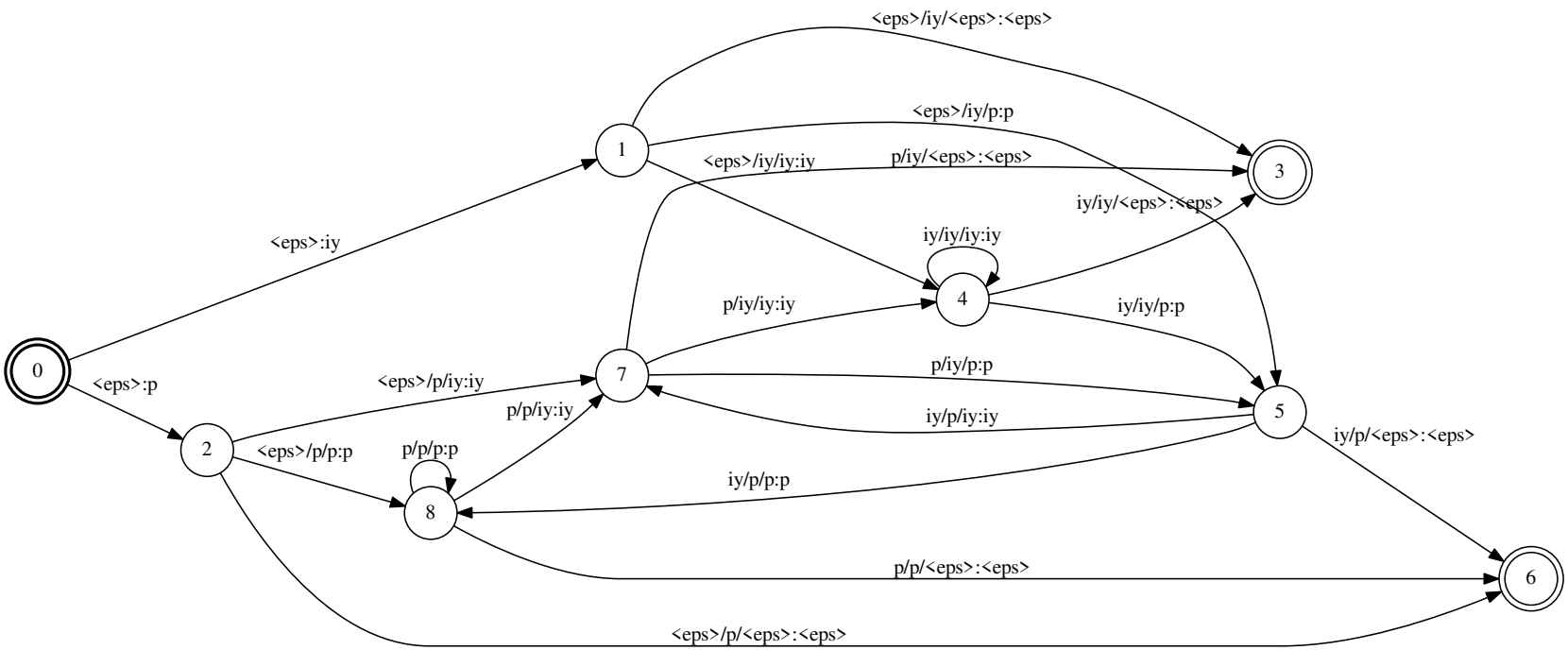
$min(det(L \circ G))$



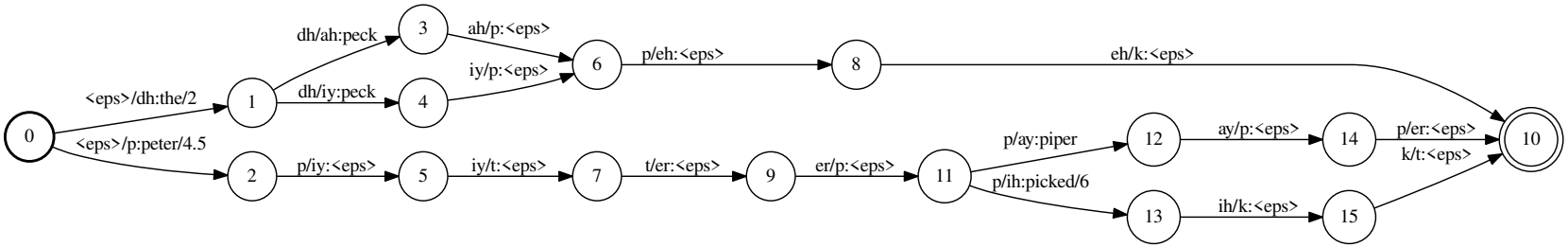
Context-dependency: left biphones



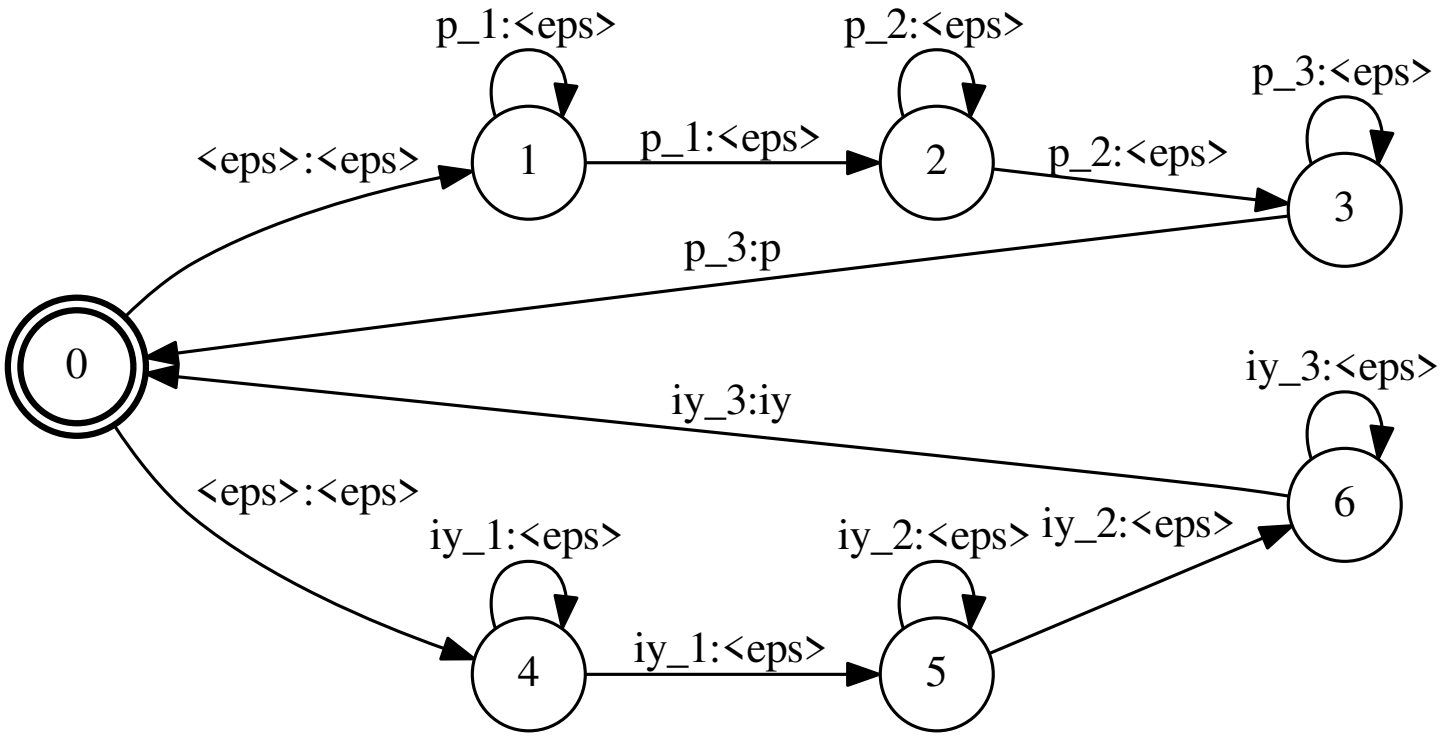
Context-dependency: triphones



$C \circ L \circ G$ – biphones



HMM transducer, H



- We can also use a version that outputs context-dependent phones
- H can be used to encode state-tying

Decoding using WFSTs

- Combining the transducers gives an overall HMM structure for the ASR system – but minimisation and determination operations on the WFSTs means it is much smaller than naively combining the HMMs
- But it is important in which order the algorithms are combined otherwise the transducers may "blow-up"
- standard approach is to determinize and minimize after each composition
- In Kaldi, ignoring one or two details

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

Reading

- Mohri et al (2008). "Speech recognition with weighted finite-state transducers." In Springer Handbook of Speech Processing, pp. 559-584. Springer.
  http://www.cs.nyu.edu/~mohri/pub/hbka.pdf
- WFSTs in Kaldi. http://danielpovey.com/files/Lecture4.pdf