WFSTs for ASR

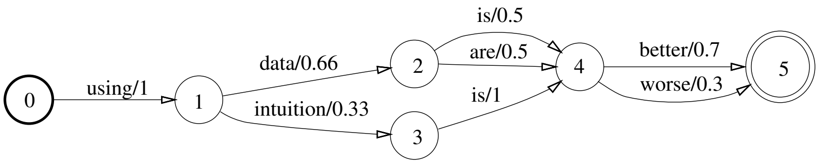
Peter Bell

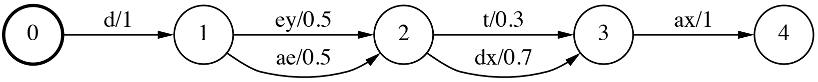
Automatic Speech Recognition – ASR Lecture 10
13 February 2020

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
- Weights use the *log semi-ring* or *tropical semi-ring* – with operations that correspond to multiplication and addition of probabilities
- There is a single start state. Any state can optionally be a final state (with a weight)
- Used by Kaldi

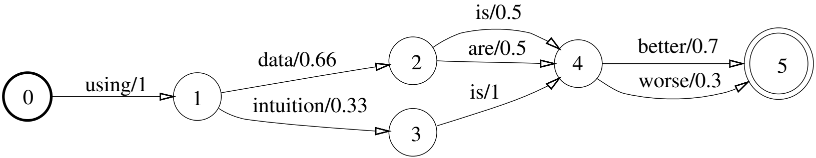
Weighted Finite State Acceptors



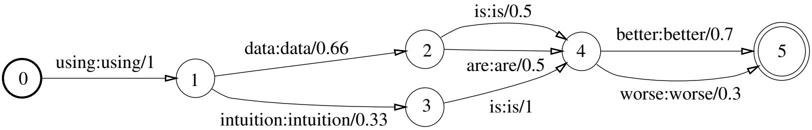


Weighted Finite State Transducers

Acceptor

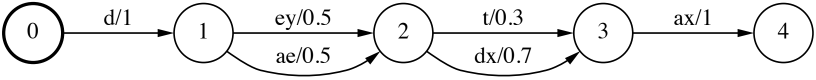


Transducer

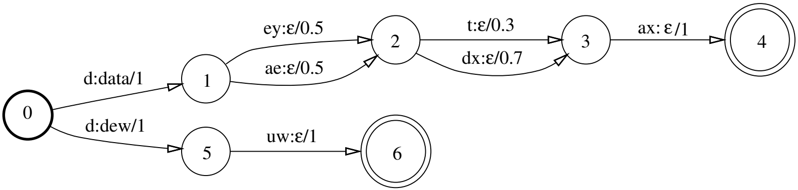


Weighted Finite State Transducers

Acceptor



Transducer



WFST Algorithms

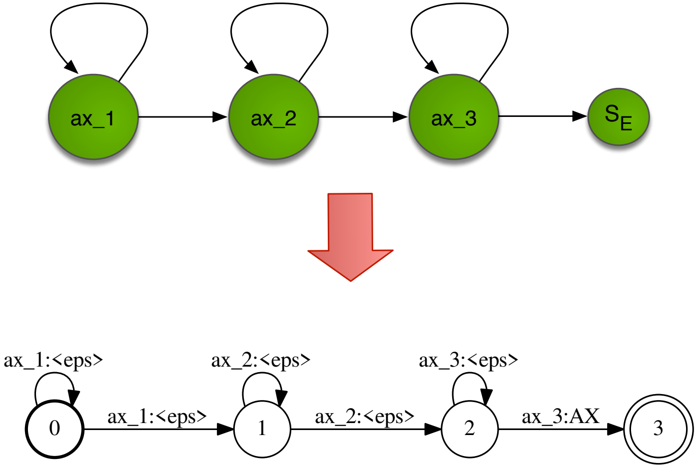
Composition Combine transducers T_1 and T_2 into a single transducer acting as if the output of T_1 was passed into T_2.

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

The HMM as a WFST



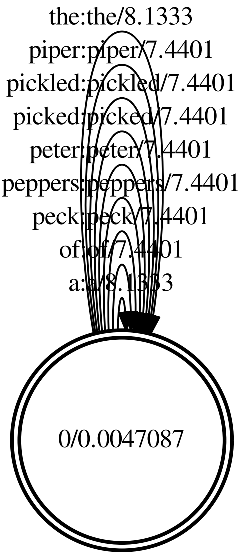
Applying WFSTs to speech recognition

- Represent the following components as WFSTs

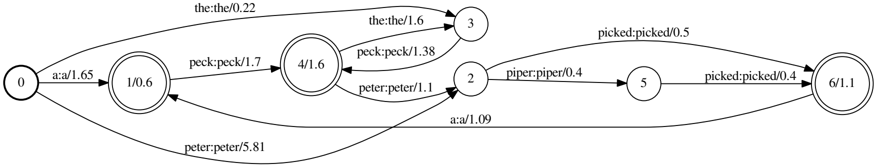
	transducer	input sequence	output sequence
G	word-level grammar	words	words
L	pronunciation lexicon	phones	words
C	context-dependency	CD phones	phones
H	HMM	HMM states	CD phones

- 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

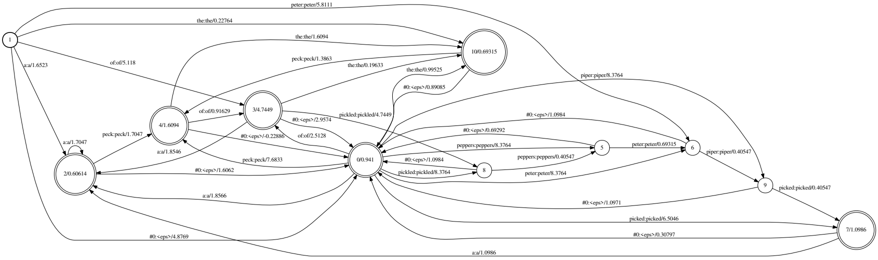
Grammar - unigram



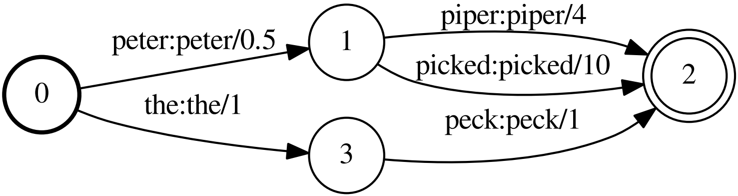
Grammar - bigram



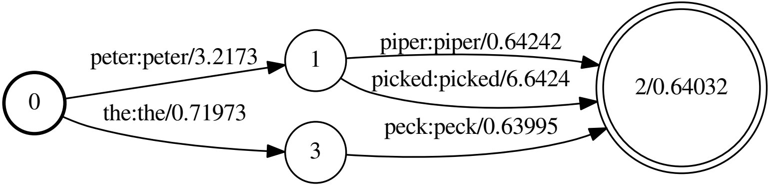
Bigram with back-off



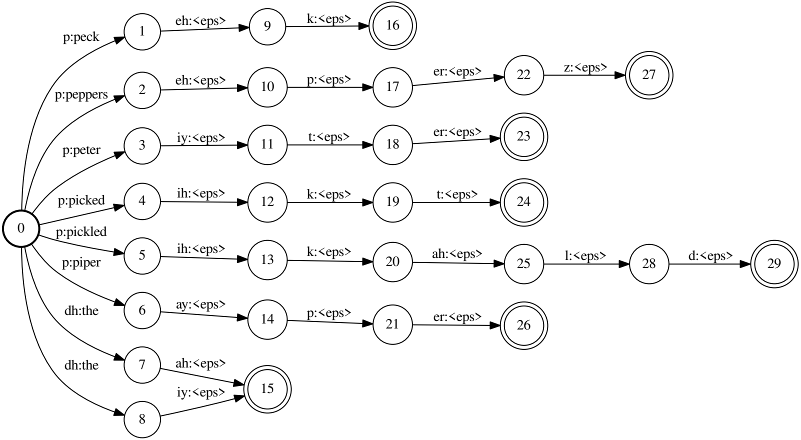
Weight pushing



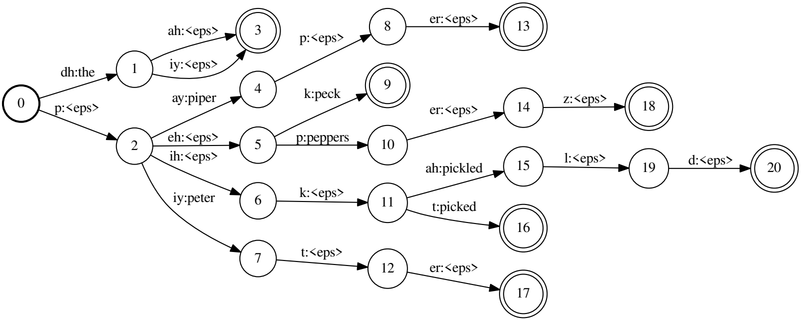
Weight-pushed version



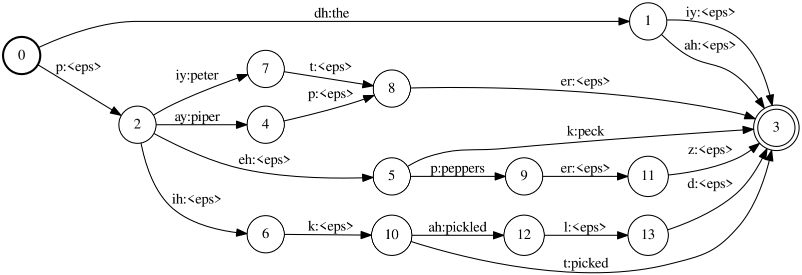
Lexicon, *L*


0
p:peck
p:peppers
p:peter
p:picked
p:pickled
p:piper
dh:the
dh:the
1
2
3
4
5
6
7
8
eh:<eps>
eh:<eps>
iy:<eps>
ih:<eps>
ih:<eps>
ay:<eps>
ah:<eps>
iy:<eps>
9
10
11
12
13
14
15
k:<eps>
p:<eps>
t:<eps>
k:<eps>
k:<eps>
p:<eps>
16
17
18
19
20
21
er:<eps>
er:<eps>
t:<eps>
ah:<eps>
er:<eps>
22
23
24
25
26
z:<eps>
l:<eps>
27
28
d:<eps>
29

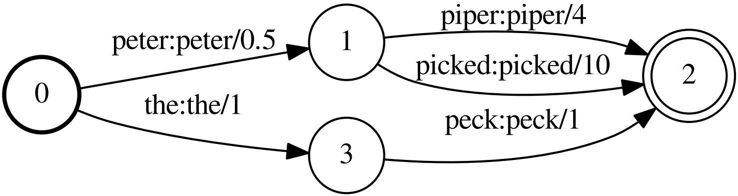
Determinization – $det(L)$



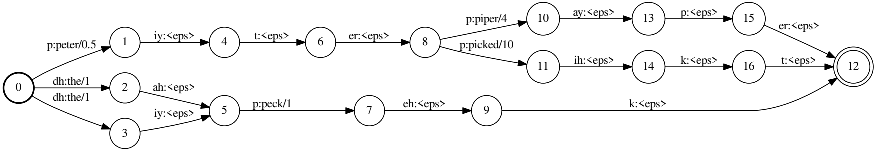
Minimization – $min(det(L))$



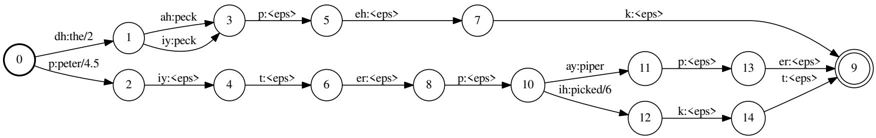
Composition



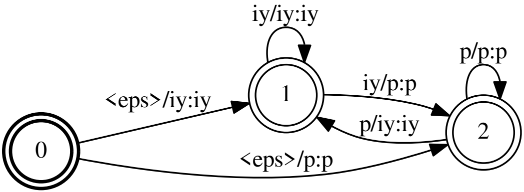
Composition: $L \circ G$



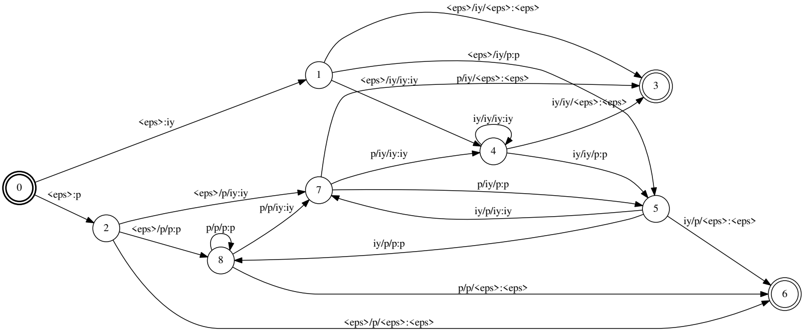
$min(det(L \circ G))$



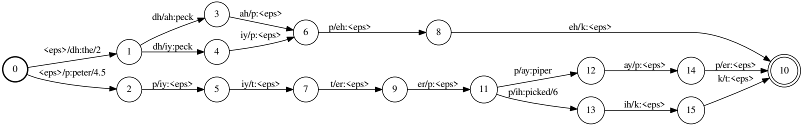
Context-dependency: biphones



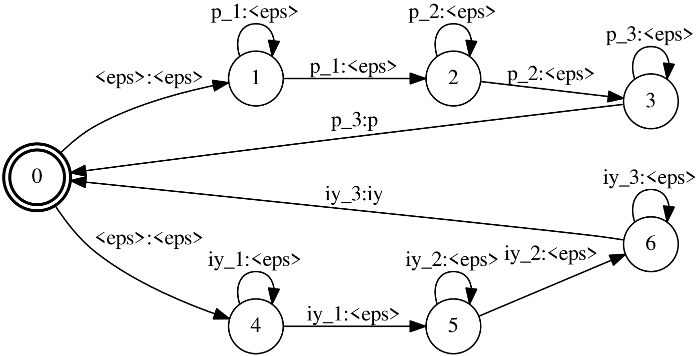
Context-dependency: triphones



$C \circ L \circ G$ – 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