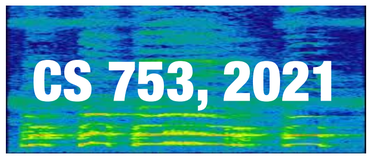
Search and Decoding

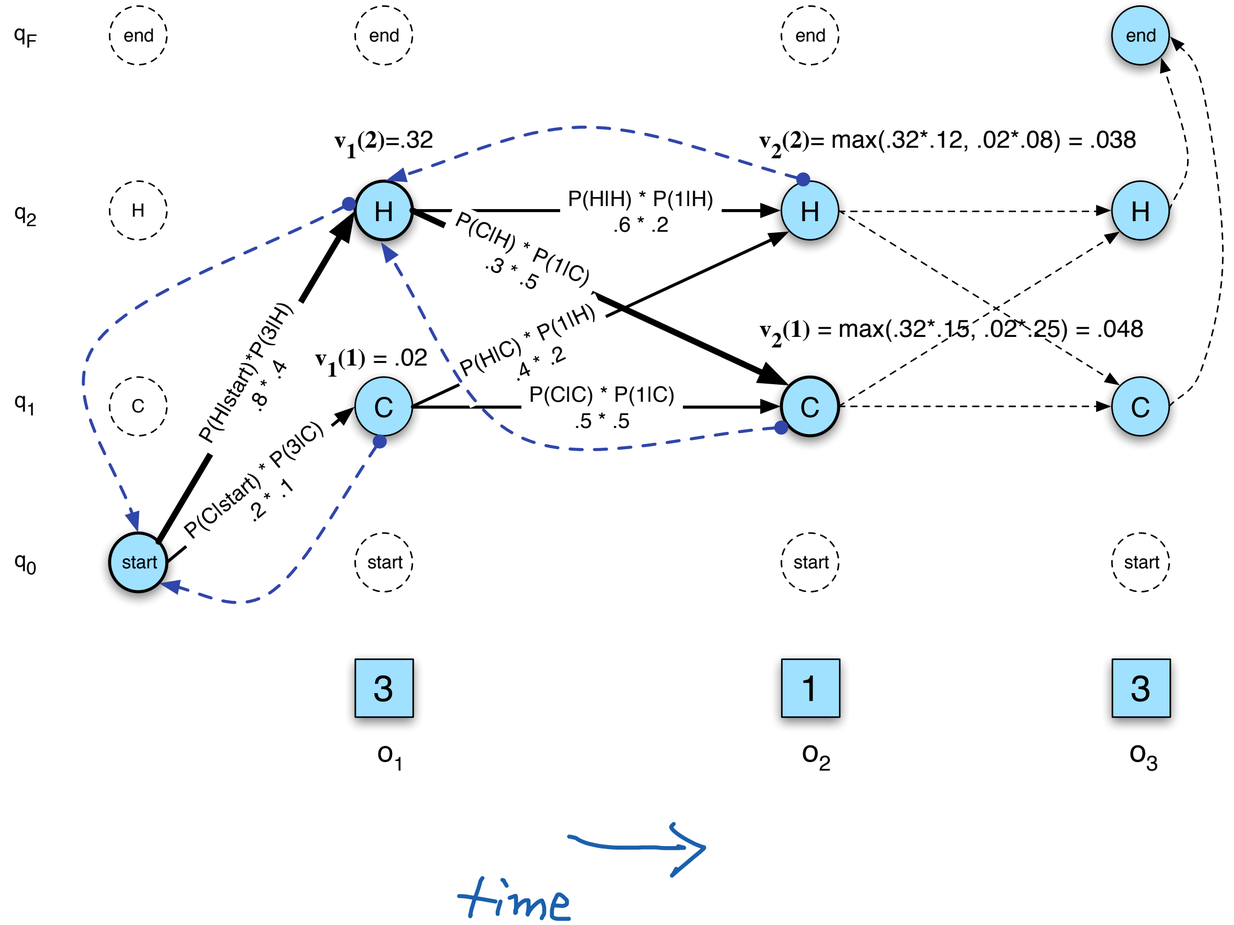
Lecture 11a



Instructor: Preethi Jyothi, IITB

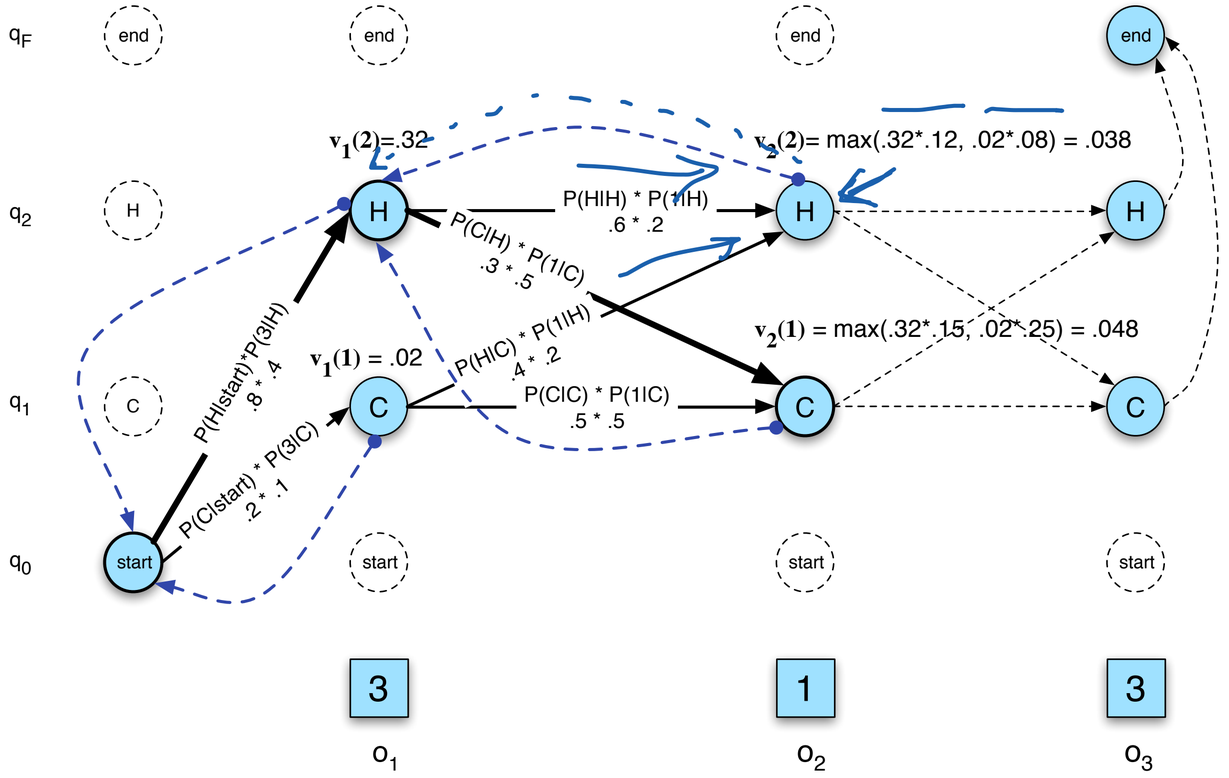
Recall Viterbi search

- Viterbi search finds the most probable path through a trellis of time on the X-axis and states on the Y-axis



Recall Viterbi search

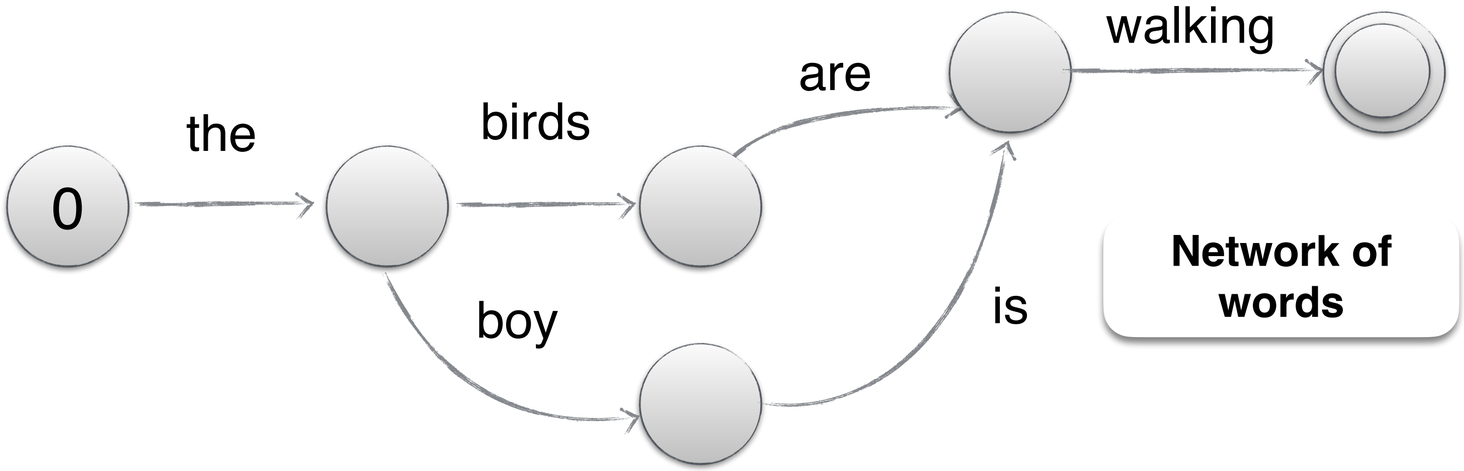
- Viterbi search finds the most probable path through a trellis of time on the X-axis and states on the Y-axis



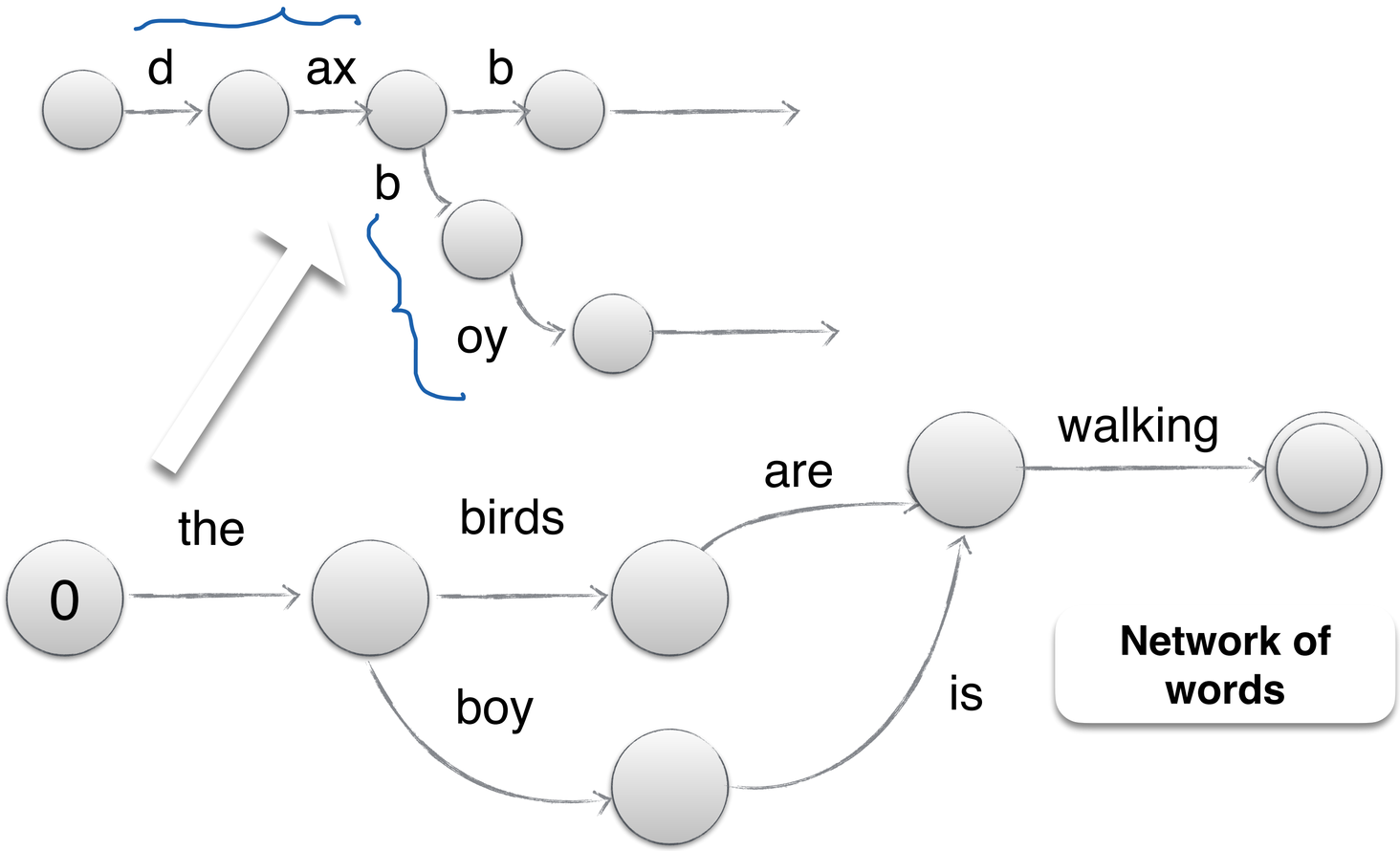
- Viterbi algorithm: Only needs to maintain information about the most probable path at each state

Image from [JM]: Jurafsky & Martin, 3rd edition, Chapter 9

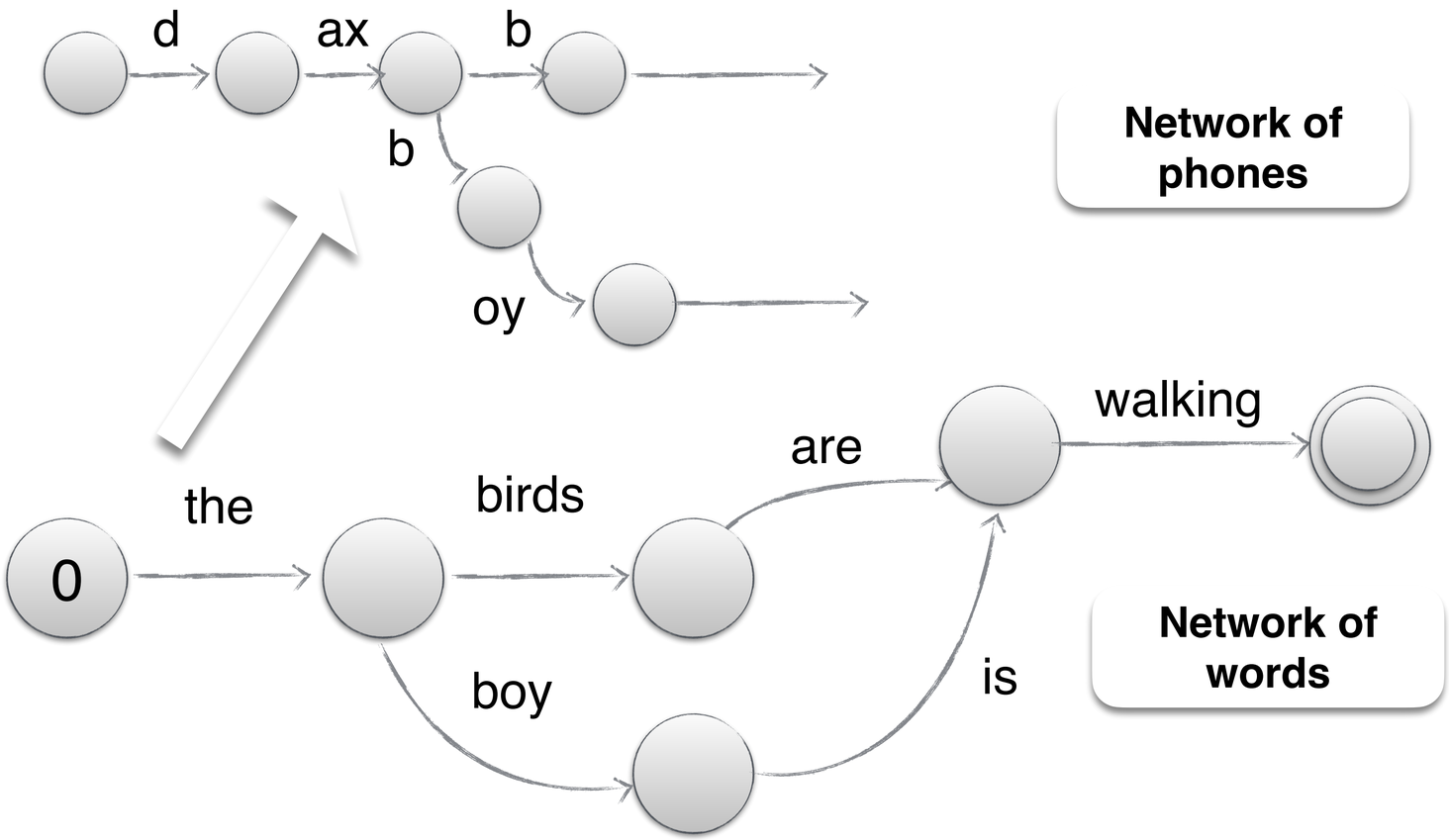
ASR Search Network



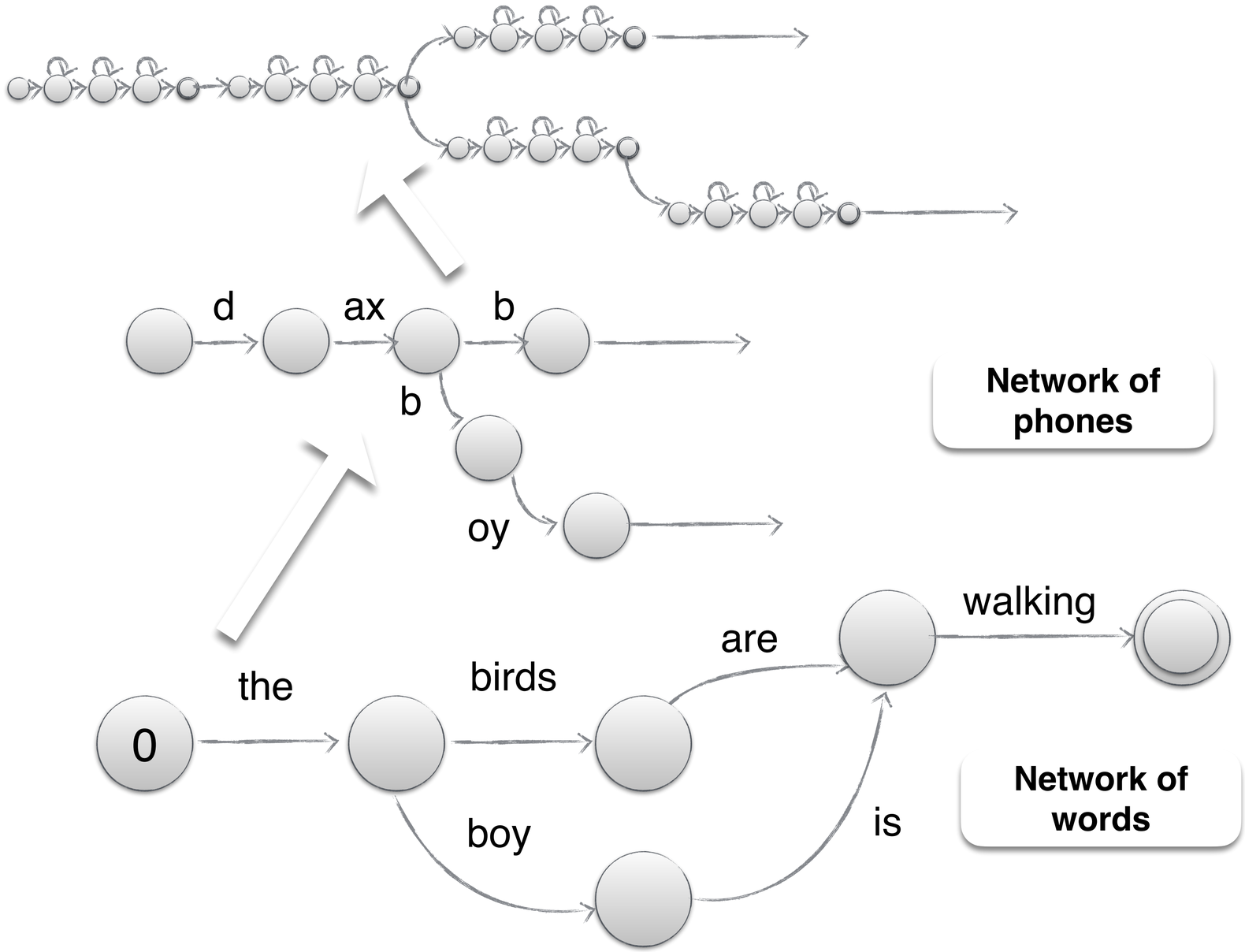
ASR Search Network



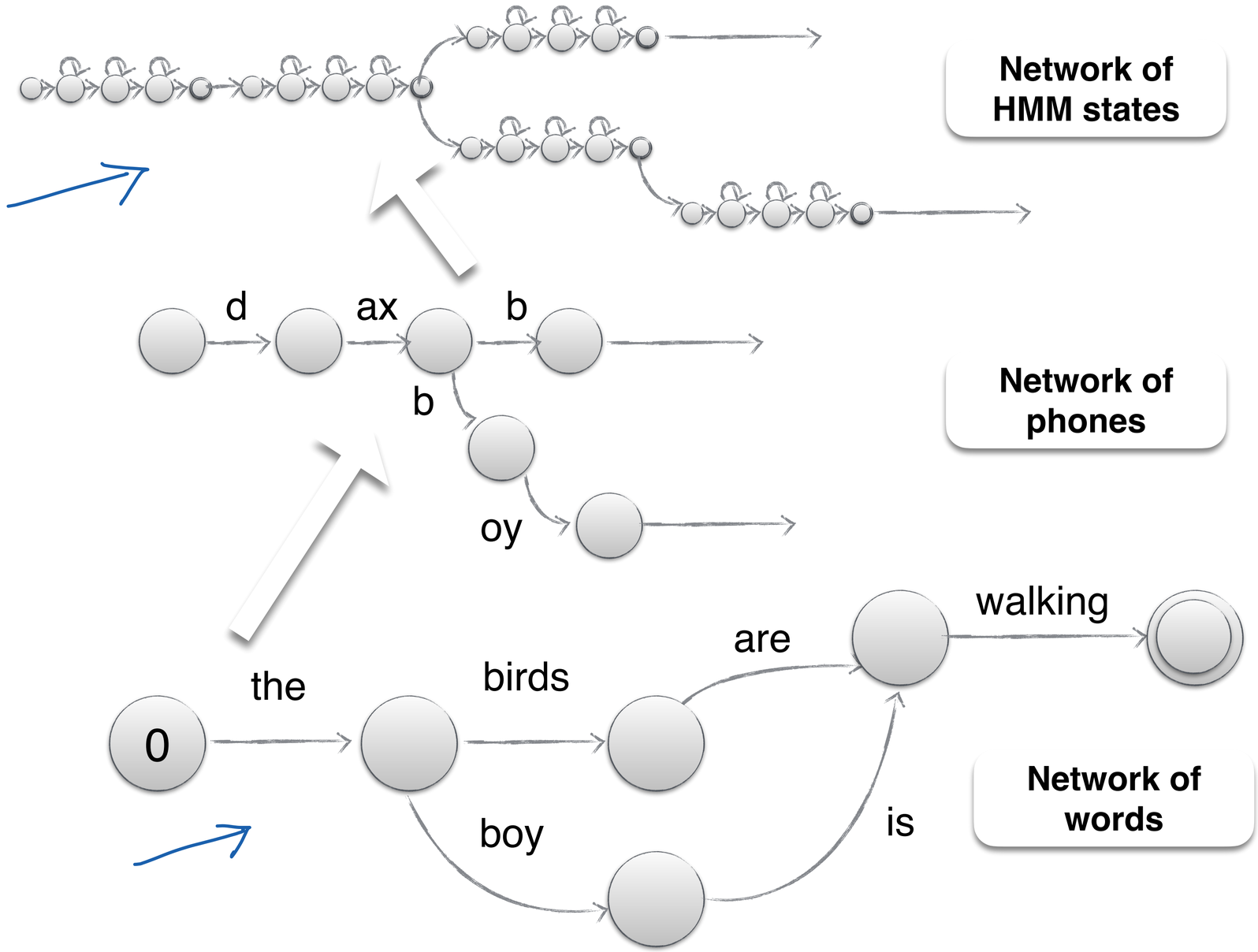
ASR Search Network



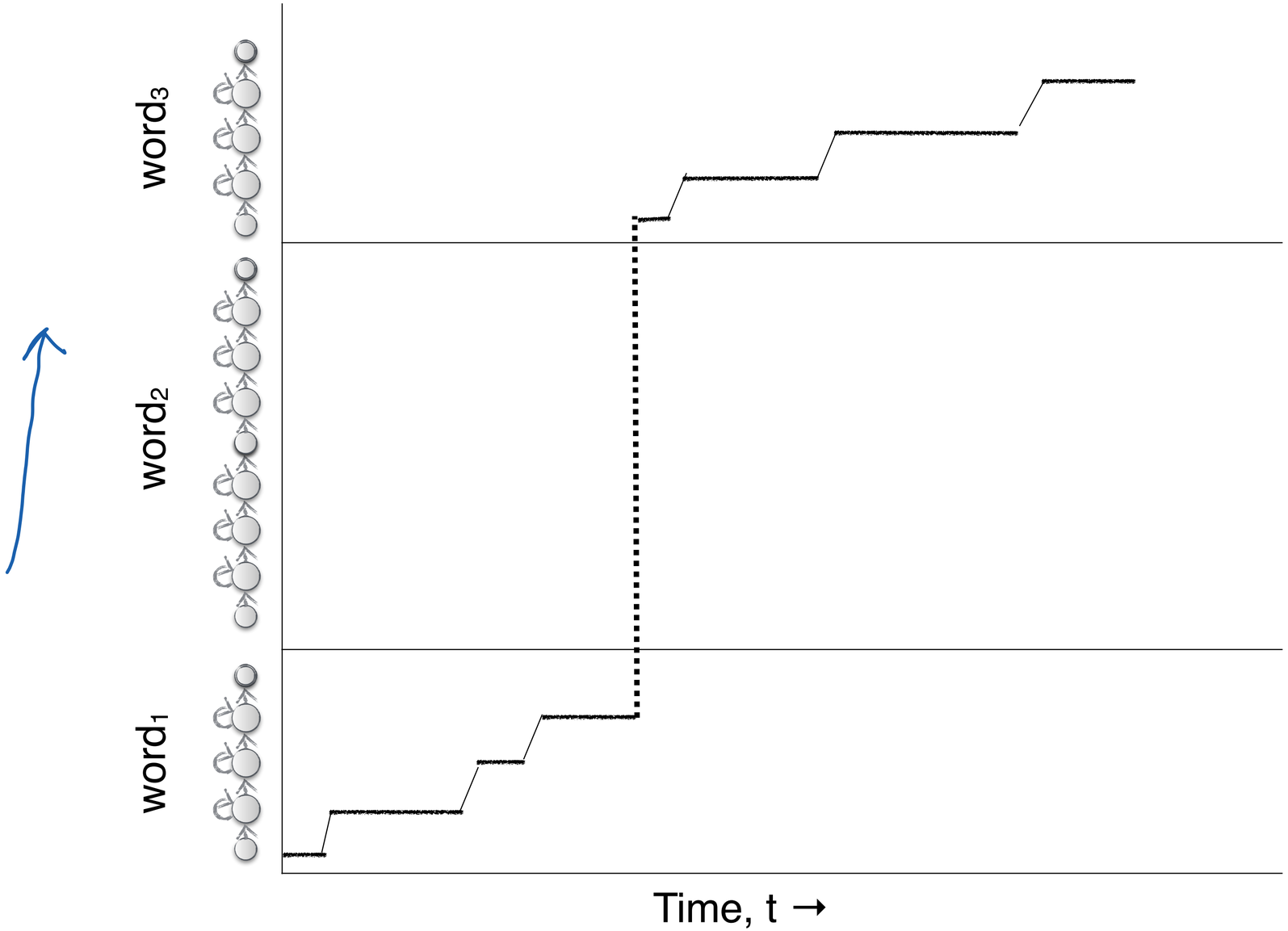
ASR Search Network



ASR Search Network



Time-state trellis



Viterbi search over the large trellis

- Exact search is infeasible for large vocabulary tasks
  - Unknown word boundaries
  - Ngram language models greatly increase the search space

Viterbi search over the large trellis

- Exact search is infeasible for large vocabulary tasks
  - Unknown word boundaries
  - Ngram language models greatly increase the search space
- Solutions
  - Compactly represent the search space using WFST-based optimisations
  - Beam search: Prune away parts of the search space that aren’t promising

Viterbi search over the large trellis

- Exact search is infeasible for large vocabulary tasks
  - Unknown word boundaries
  - Ngram language models greatly increase the search space
- Solutions
  - Compactly represent the search space using WFST-based optimisations
  - Beam search: Prune away parts of the search space that aren’t promising

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy

Recall not all weighted transducers are determinizable

To ensure determinizability of L ○ G, introduce disambiguation symbols in L to deal with homophones in the lexicon

read : r eh d #1
red : r eh d #2

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy

Recall not all weighted transducers are determinizable

To ensure determinizability of L ○ G, introduce disambiguation symbols in L to deal with homophones in the lexicon

read : r eh d #1
red : r eh d #2

$H \circ C \circ L \circ G$

Propagate the disambiguation symbols as self-loops back to C and H. Resulting machines are $\tilde{H}$, $\tilde{C}$, $\tilde{L}$

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Final optimization cascade:

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Final optimization cascade:

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Final optimization cascade:

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Final optimization cascade:

$$N = \pi_\varepsilon(\min(\det(\tilde{H} \circ \det(\tilde{C} \circ \det(\tilde{L} \circ G)))))$$

Two main WFST Optimizations

- Use determinization to reduce/eliminate redundancy
- Use minimization to reduce space requirements

Minimization ensures that the final composed machine has minimum number of states

Final optimization cascade:

$$N = \pi_\varepsilon(\min(\det(\tilde{H} \circ \det(\tilde{C} \circ \det(\tilde{L} \circ G)))))$$

Replaces disambiguation symbols in input alphabet of $\tilde{H}$ with ε

1st pass recognition networks (40K vocab)

transducer	x real-time
$C \circ L \circ G$	12.5
$C \circ det(L \circ G)$	1.2
$det(H \circ C \circ L \circ G)$	1.0

Recognition speeds for systems with an accuracy of 83%

Static and dynamic networks

- What we've seen so far: *Static* decoding graph
  - H ○ C ○ L ○ G
  - Determinize/minimize to make this graph more compact

Static and dynamic networks

- What we've seen so far: *Static* decoding graph
  - H ○ C ○ L ○ G
  - Determinize/minimize to make this graph more compact
- Another approach: *Dynamic* graph expansion
  - Dynamically build the graph with active states on the fly
  - Do on-the-fly composition with the language model G
    - (H ○ C ○ L) ○ G

Viterbi search over the large trellis

- Exact search is infeasible for large vocabulary tasks
  - Unknown word boundaries
  - Ngram language models greatly increase the search space
- Solutions
  - Compactly represent the search space using WFST-based optimisations
  - Beam search: Prune away parts of the search space that aren’t promising

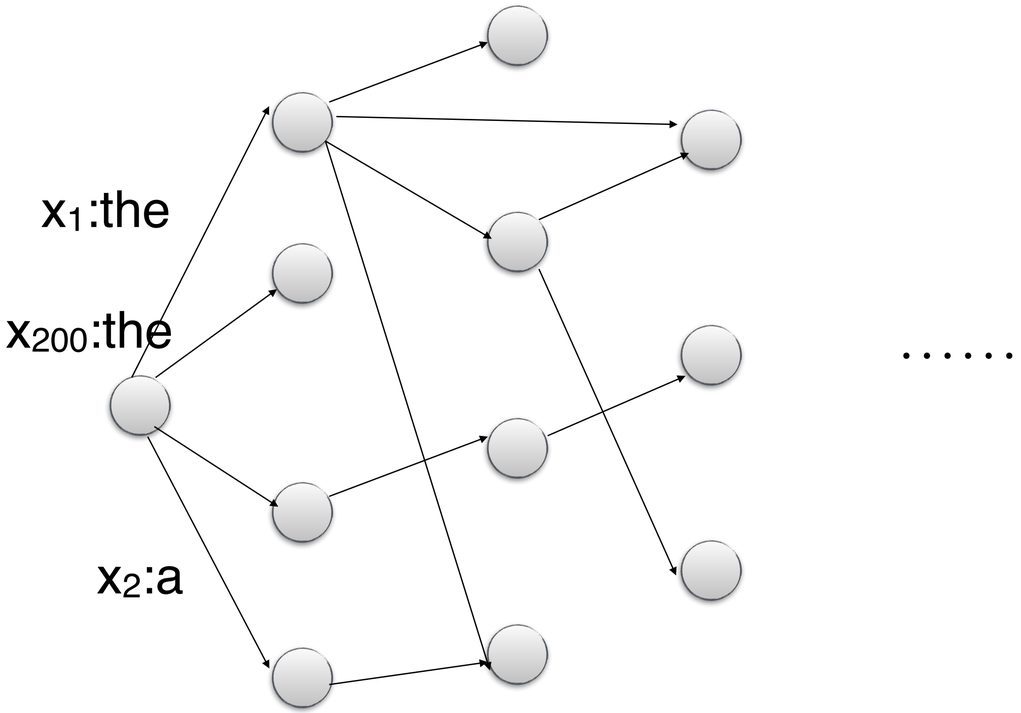
Beam search algorithm

Initialization: c*urrent states* := *initial state*

while (*current states* do not contain the *goal state*) do:

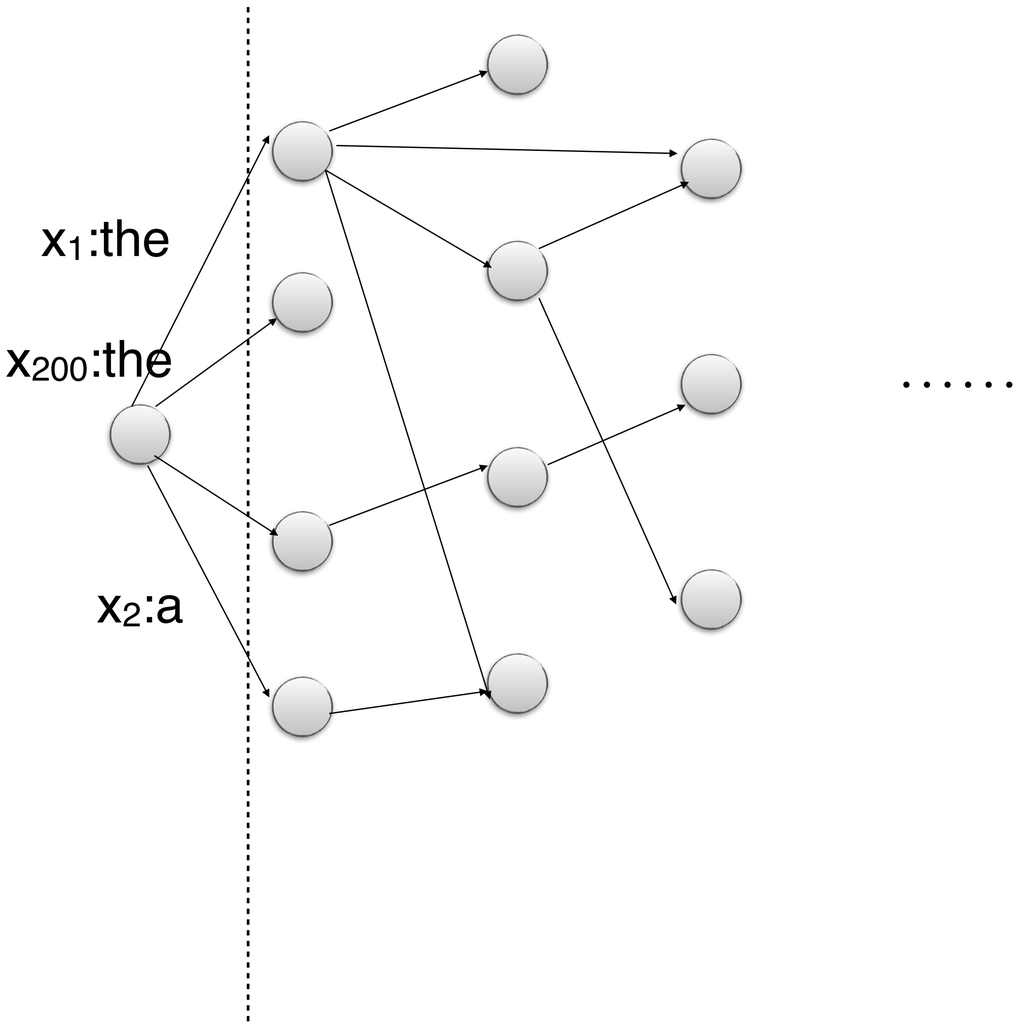
> *successor states* := NEXT(*current states*)
>
> where NEXT is next state function
>
> score the *successor states*
>
> set *current states* to a pruned set of *successor states* using beam width δ
>
> only retain those *successor states* that are within a fixed δ of the score of the best path

Beam search over the decoding graph



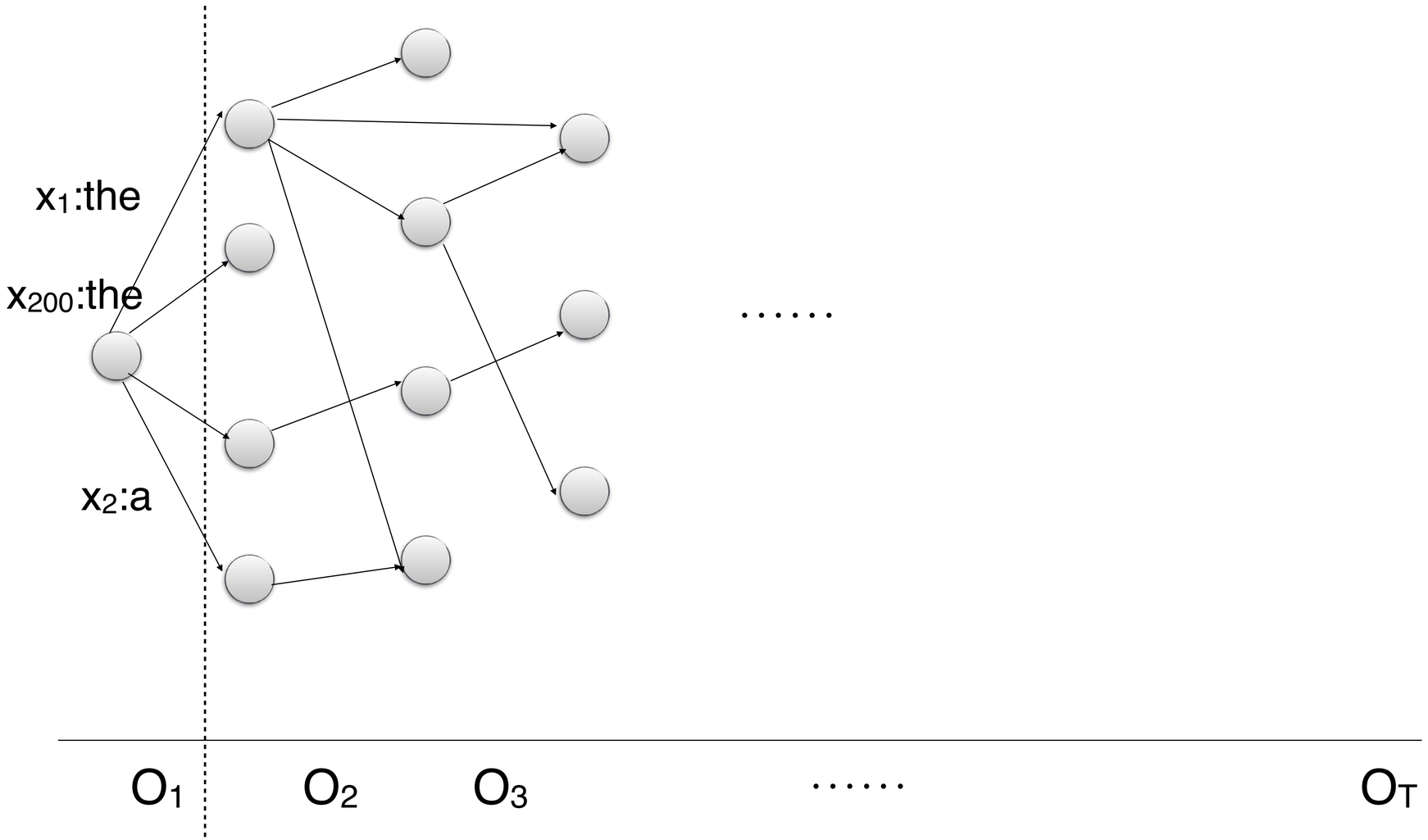
.....

Beam search over the decoding graph

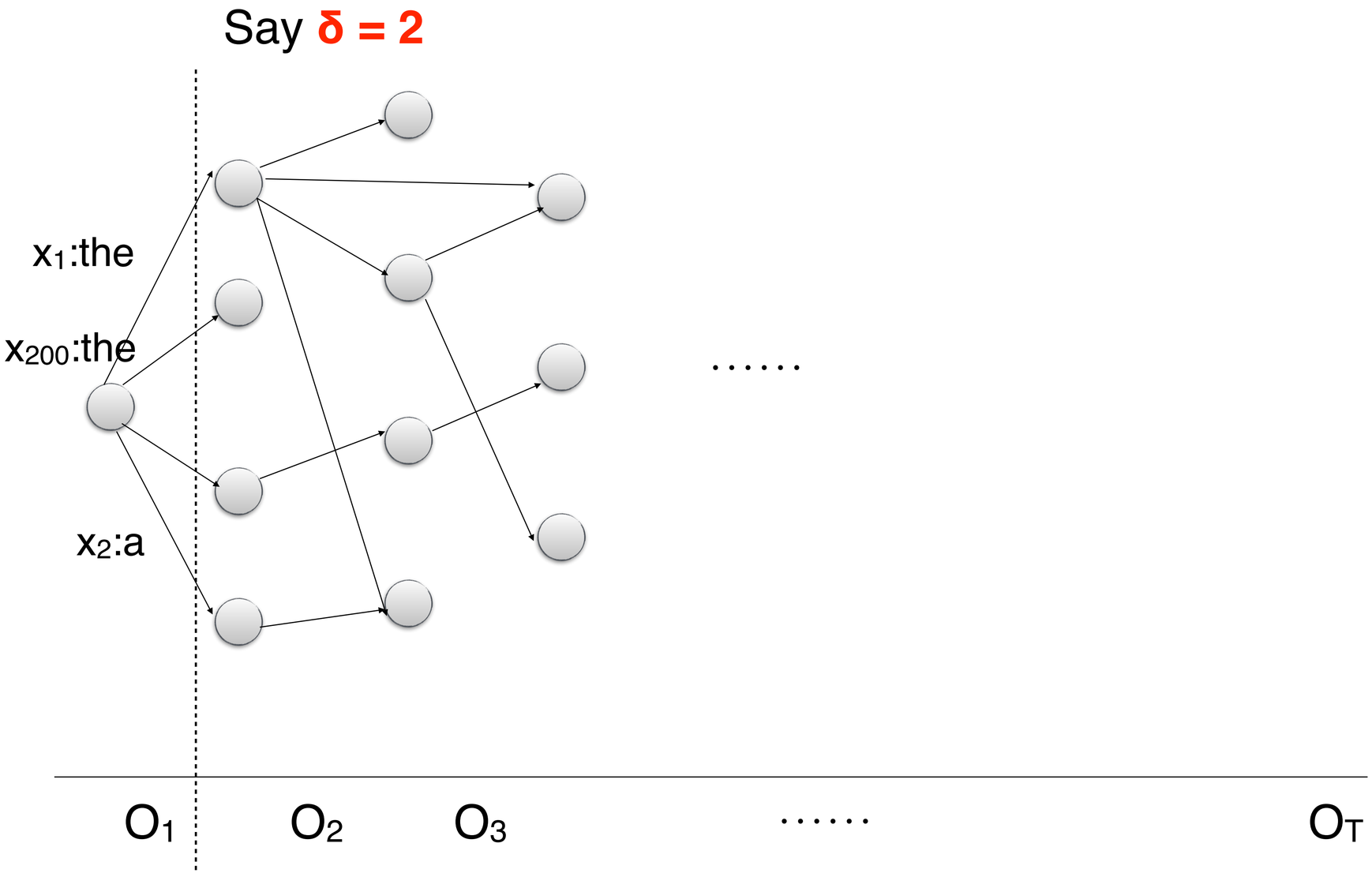


.....

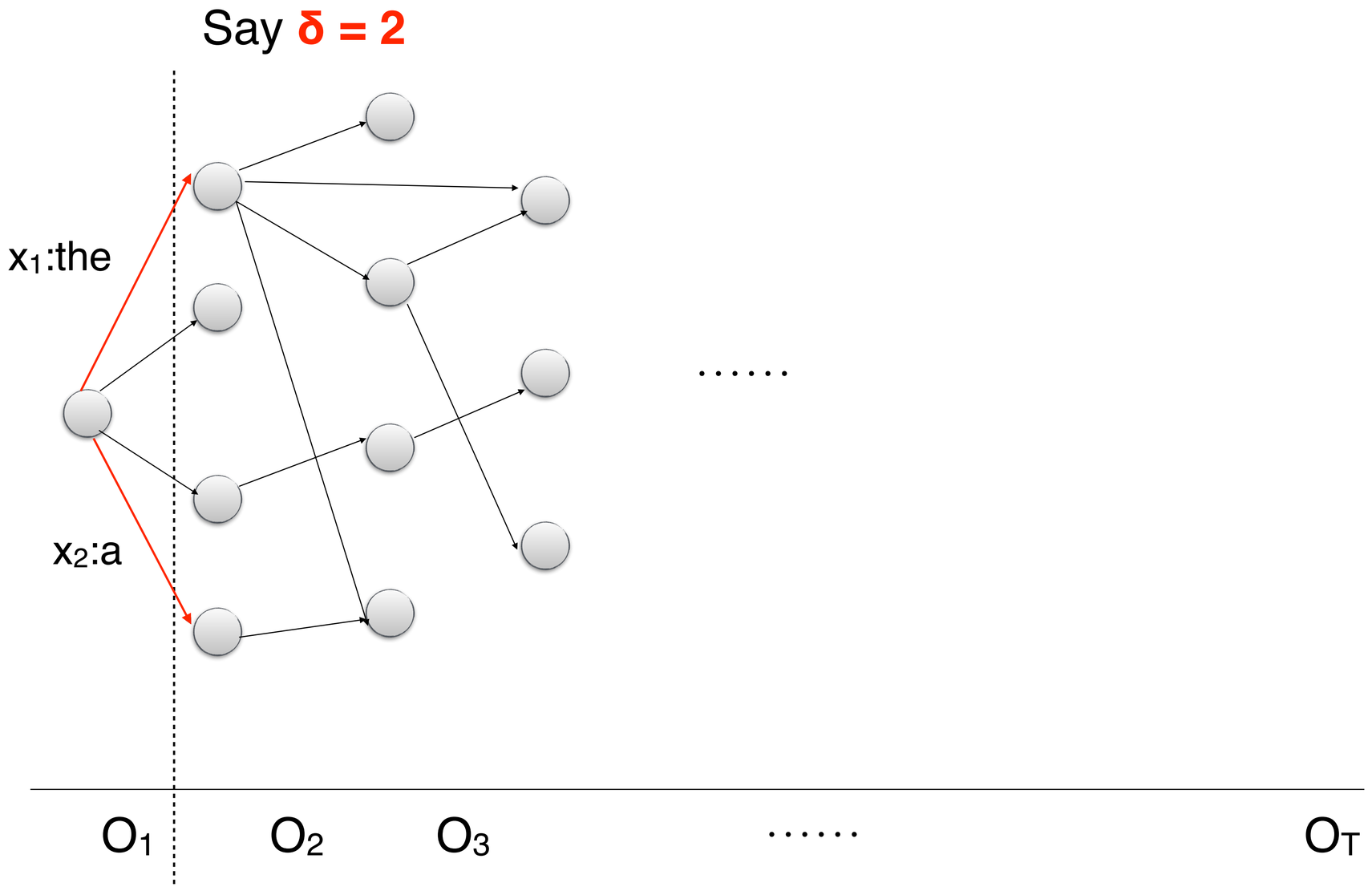
Beam search over the decoding graph



Beam search over the decoding graph



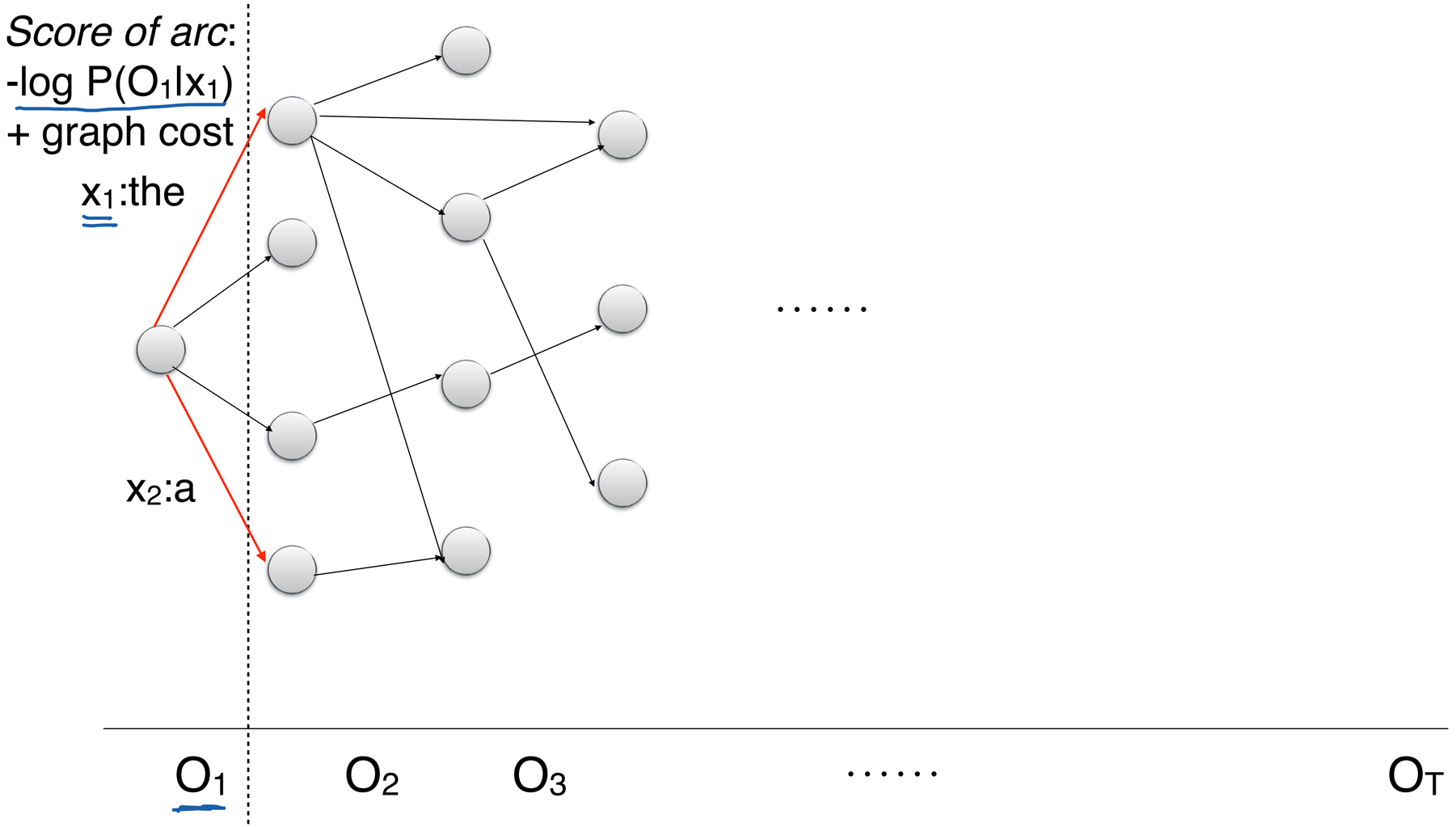
Beam search over the decoding graph



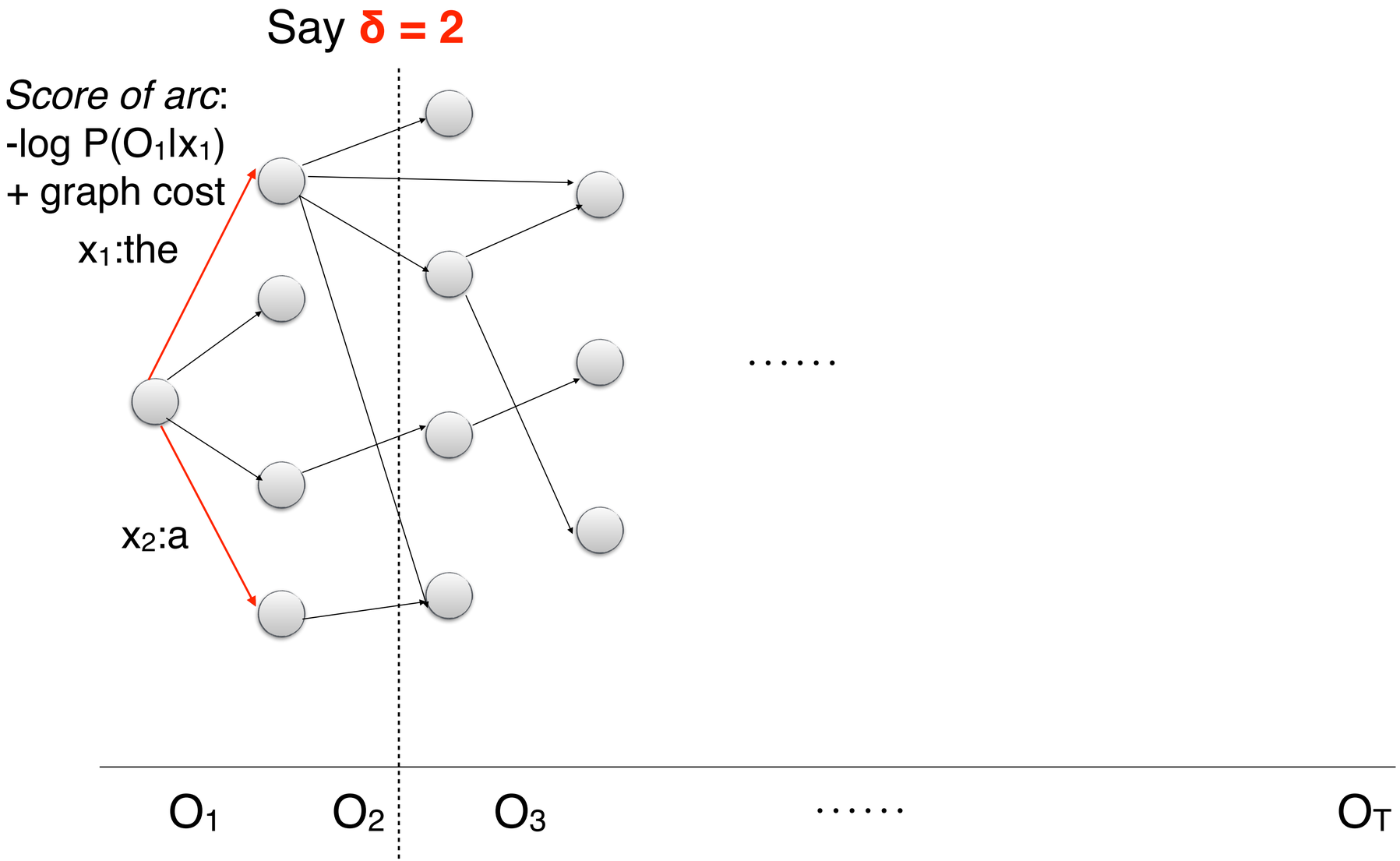
Beam search over the decoding graph

$H \circ C \circ L \circ G$

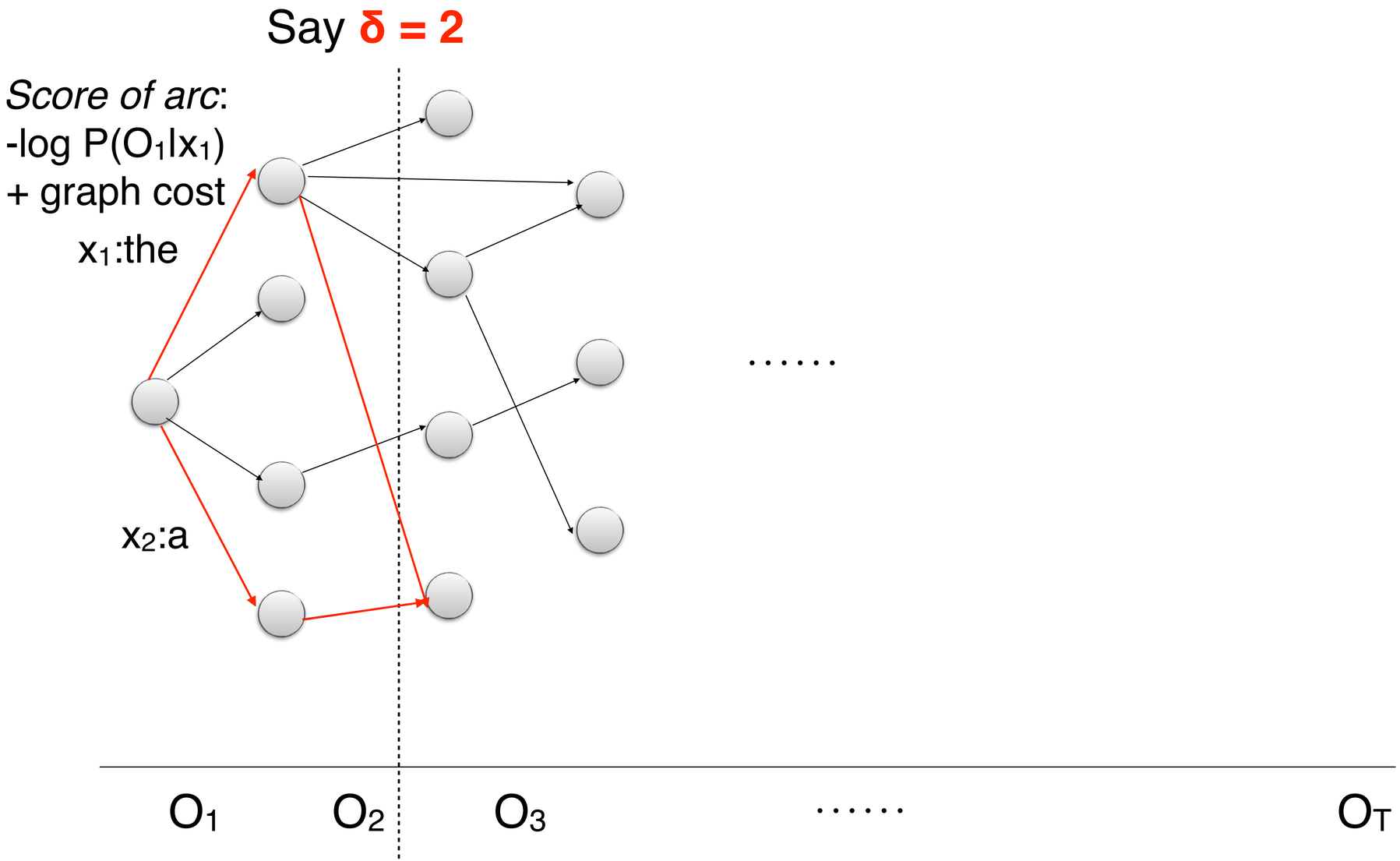
Say **δ = 2**



Beam search over the decoding graph



Beam search over the decoding graph



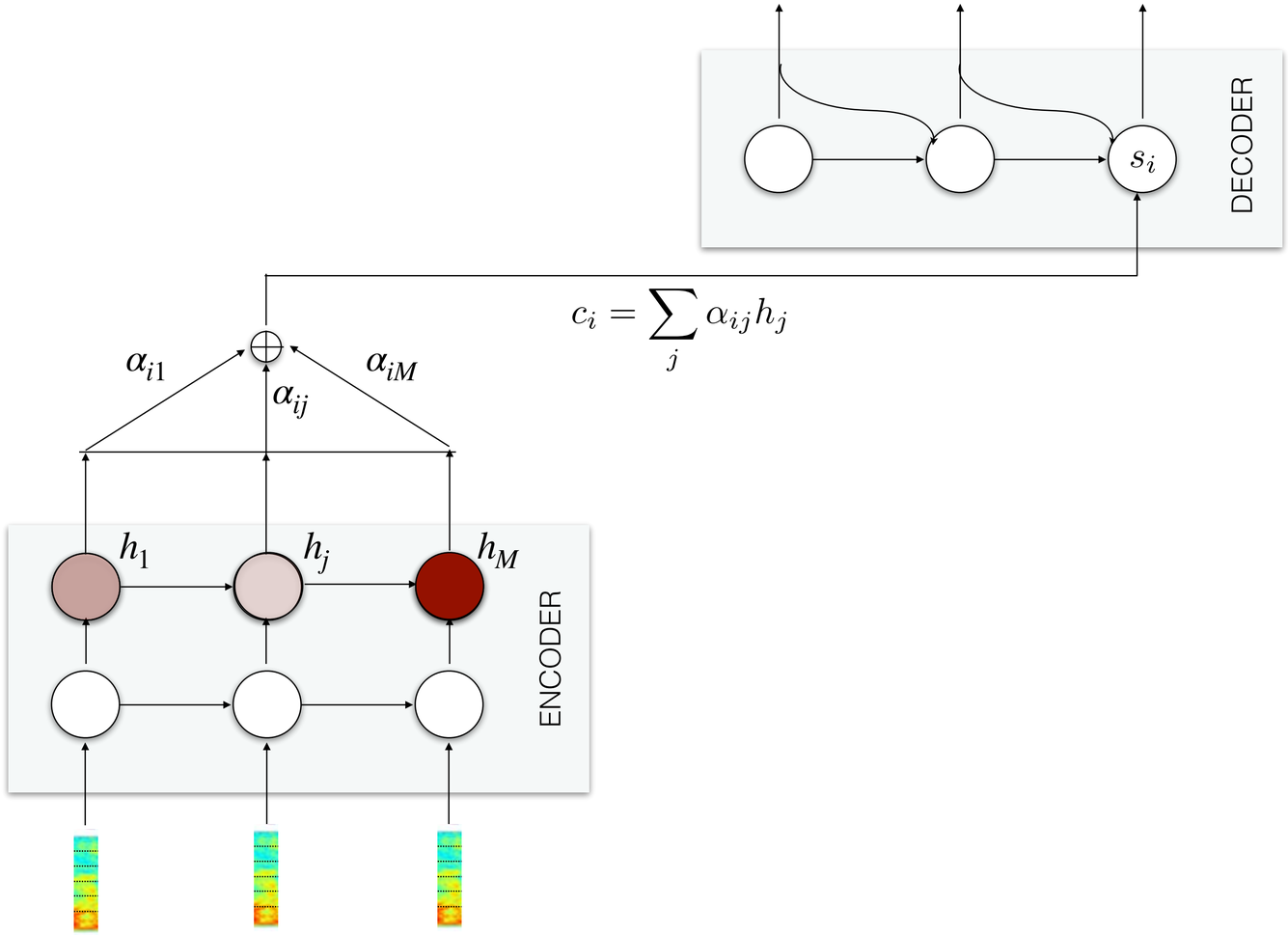
Beam pruning

- At each time-step t, only retain those nodes in the time-state trellis that are within a fixed threshold δ (beam width) of the best path

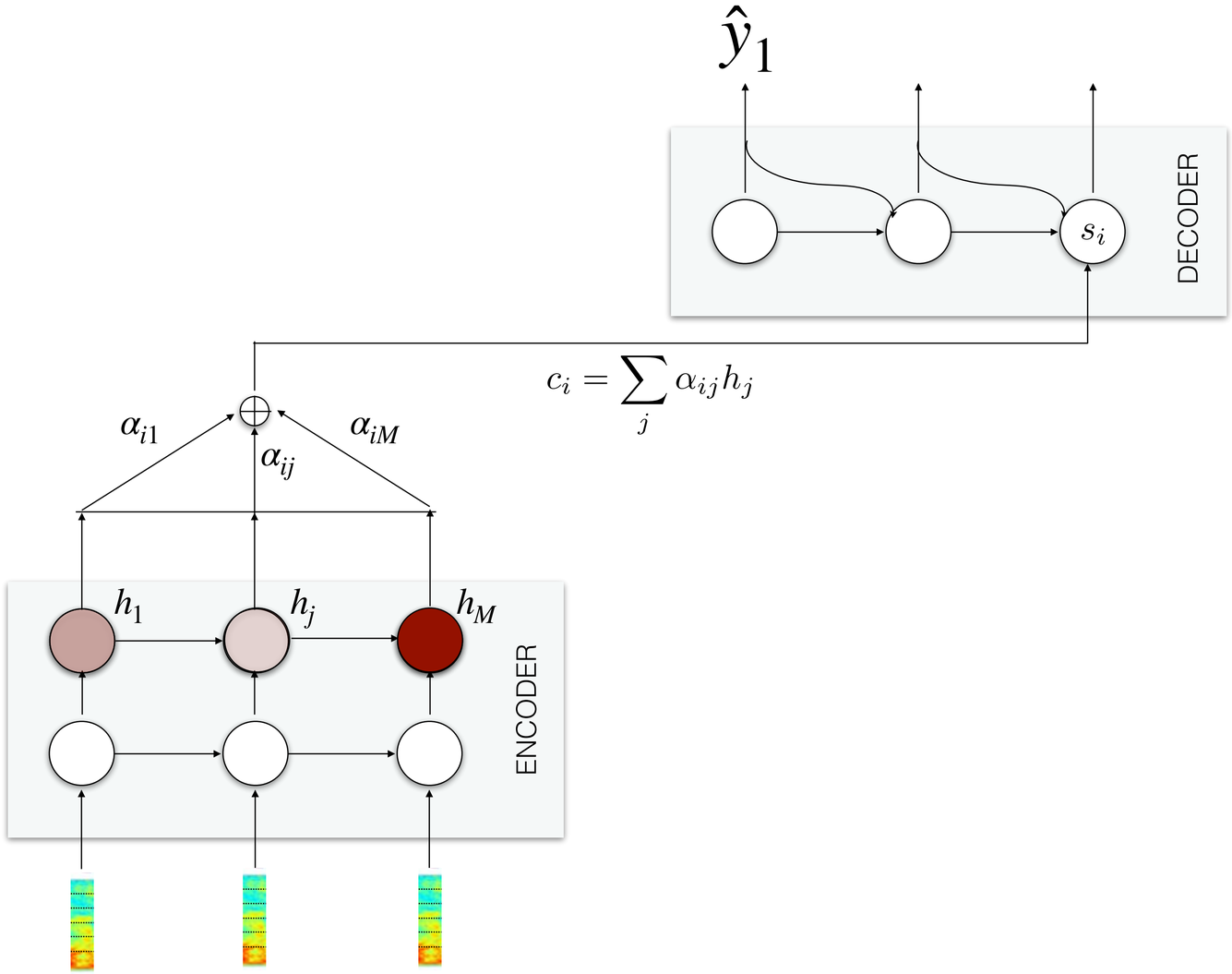
Beam pruning

- At each time-step t, only retain those nodes in the time-state trellis that are within a fixed threshold δ (beam width) of the best path
- Given active nodes from the last time-step:
  - Examine nodes in the current time-step …
  - … that are reachable from active nodes in the previous time-step
  - Get active nodes for the current time-step by only retaining nodes with hypotheses that score close to the score of the best hypothesis

Beam search in a seq2seq model

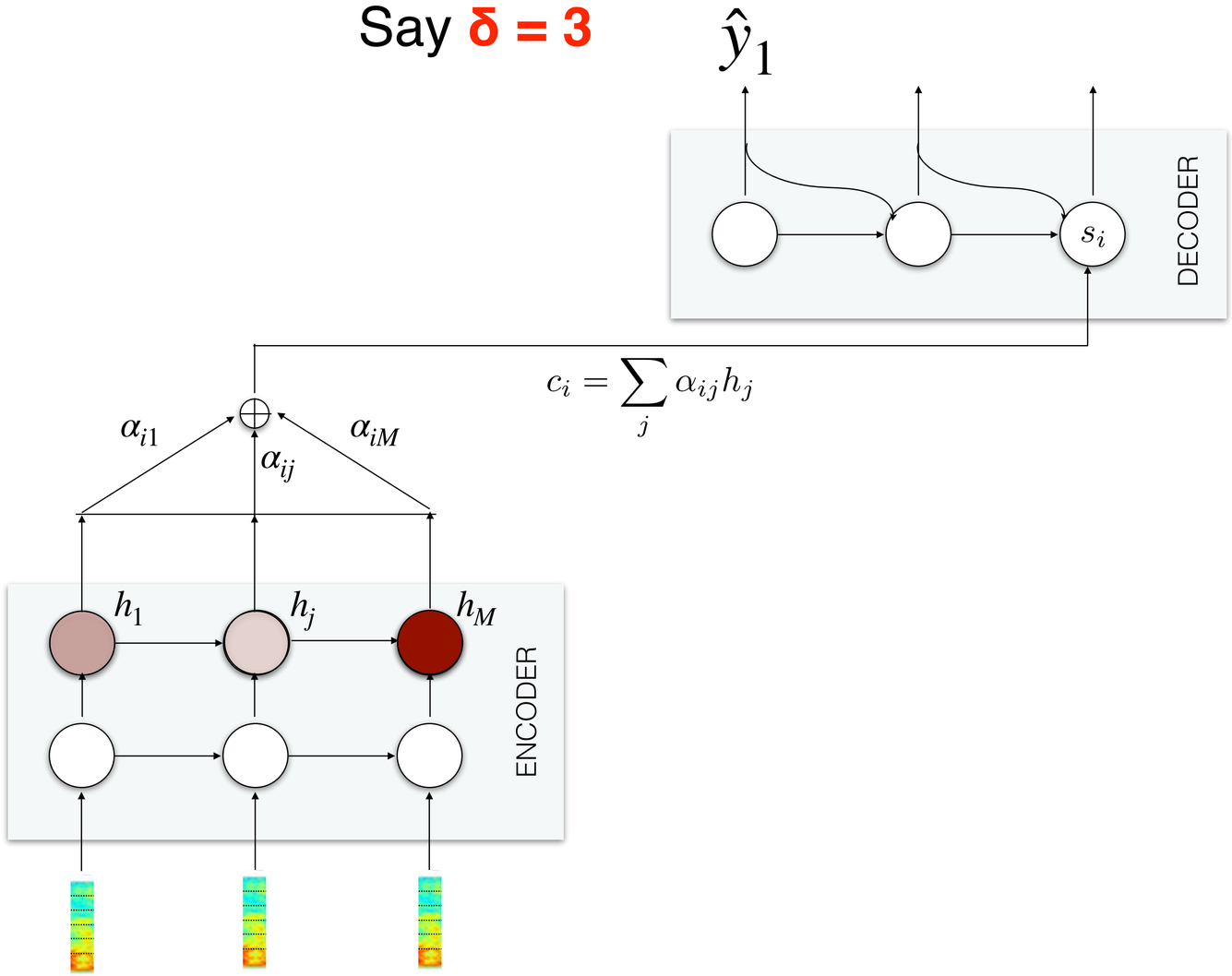


Beam search in a seq2seq model



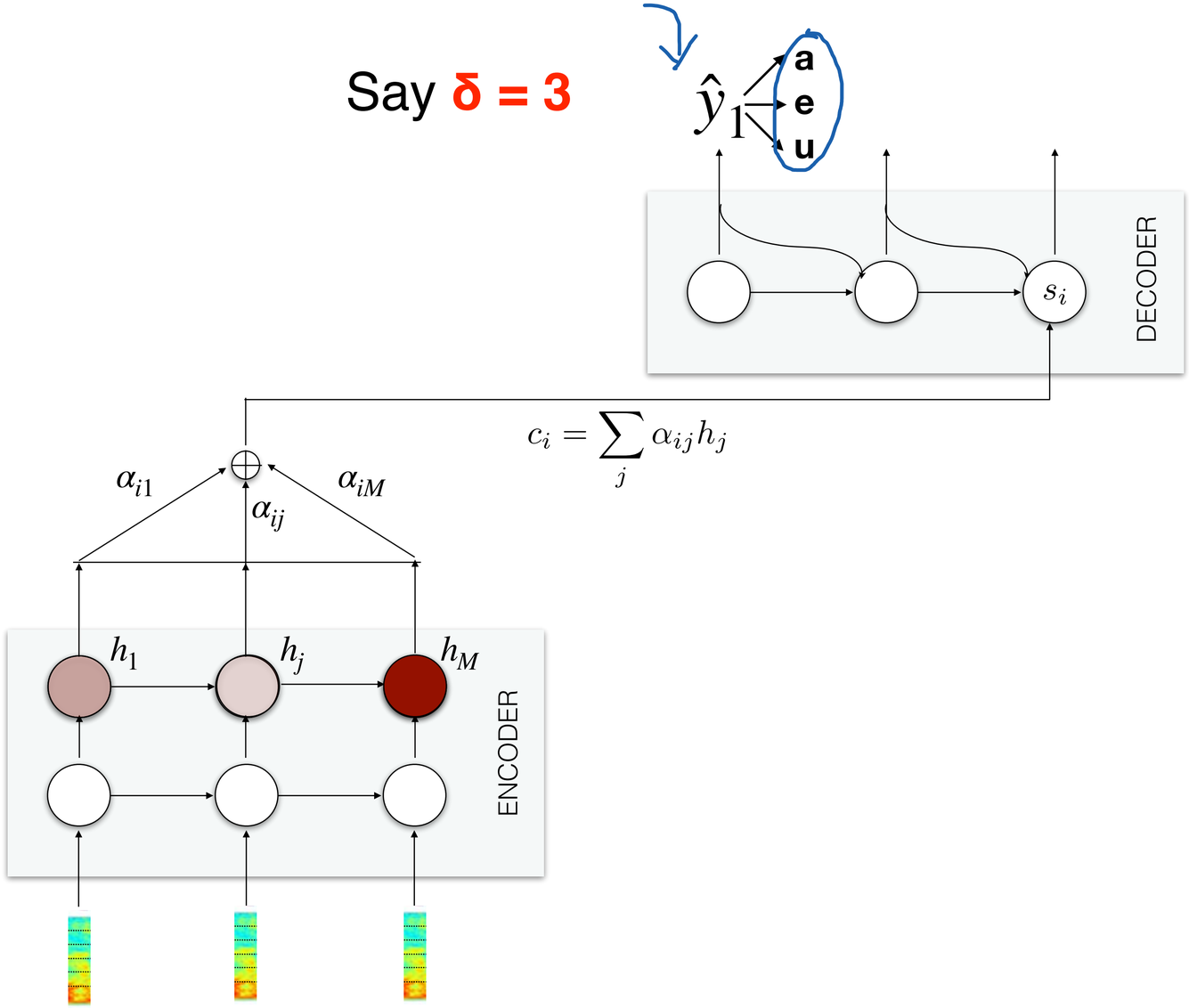
Beam search in a seq2seq model

Say **δ = 3**



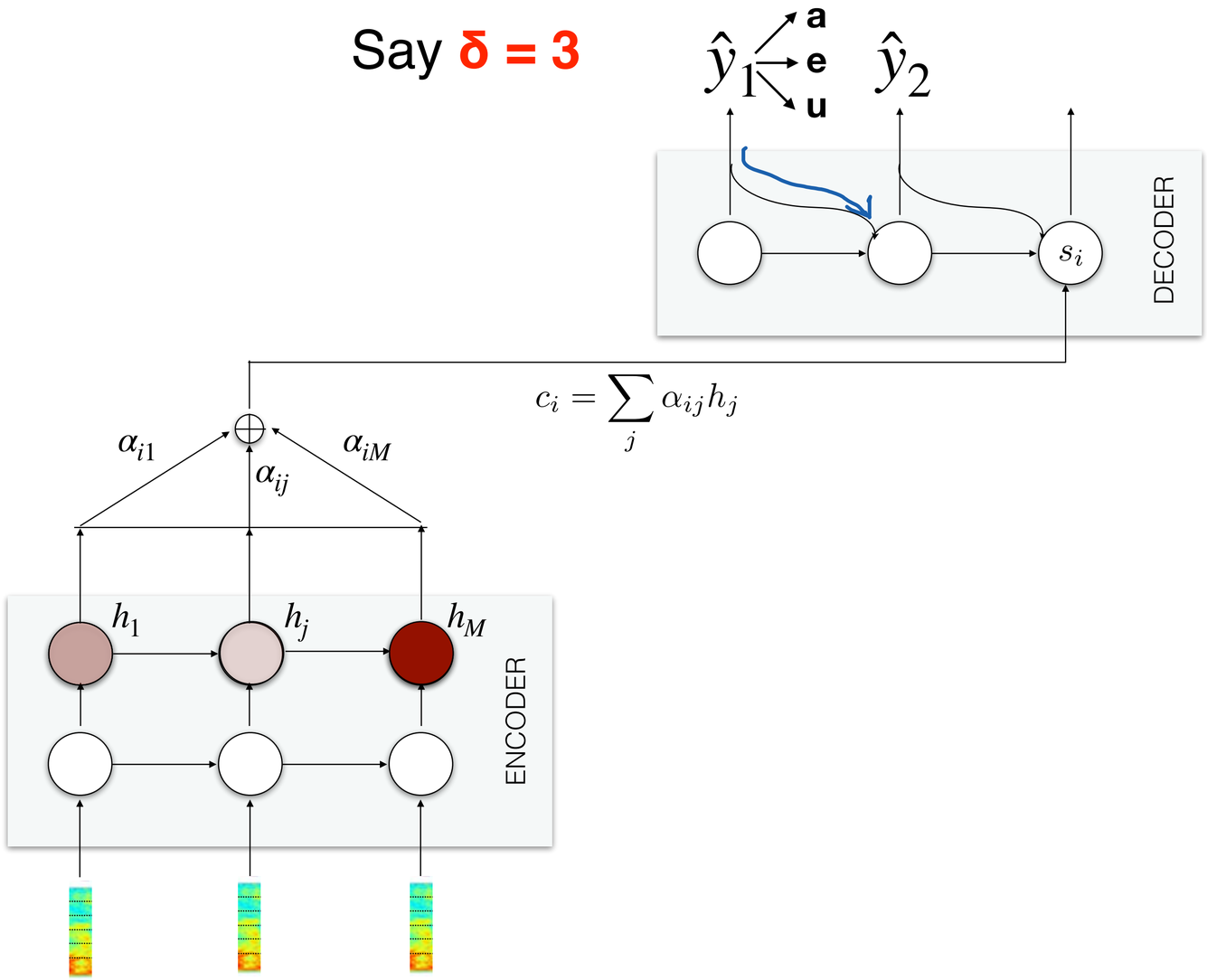
Beam search in a seq2seq model

Say **δ = 3**



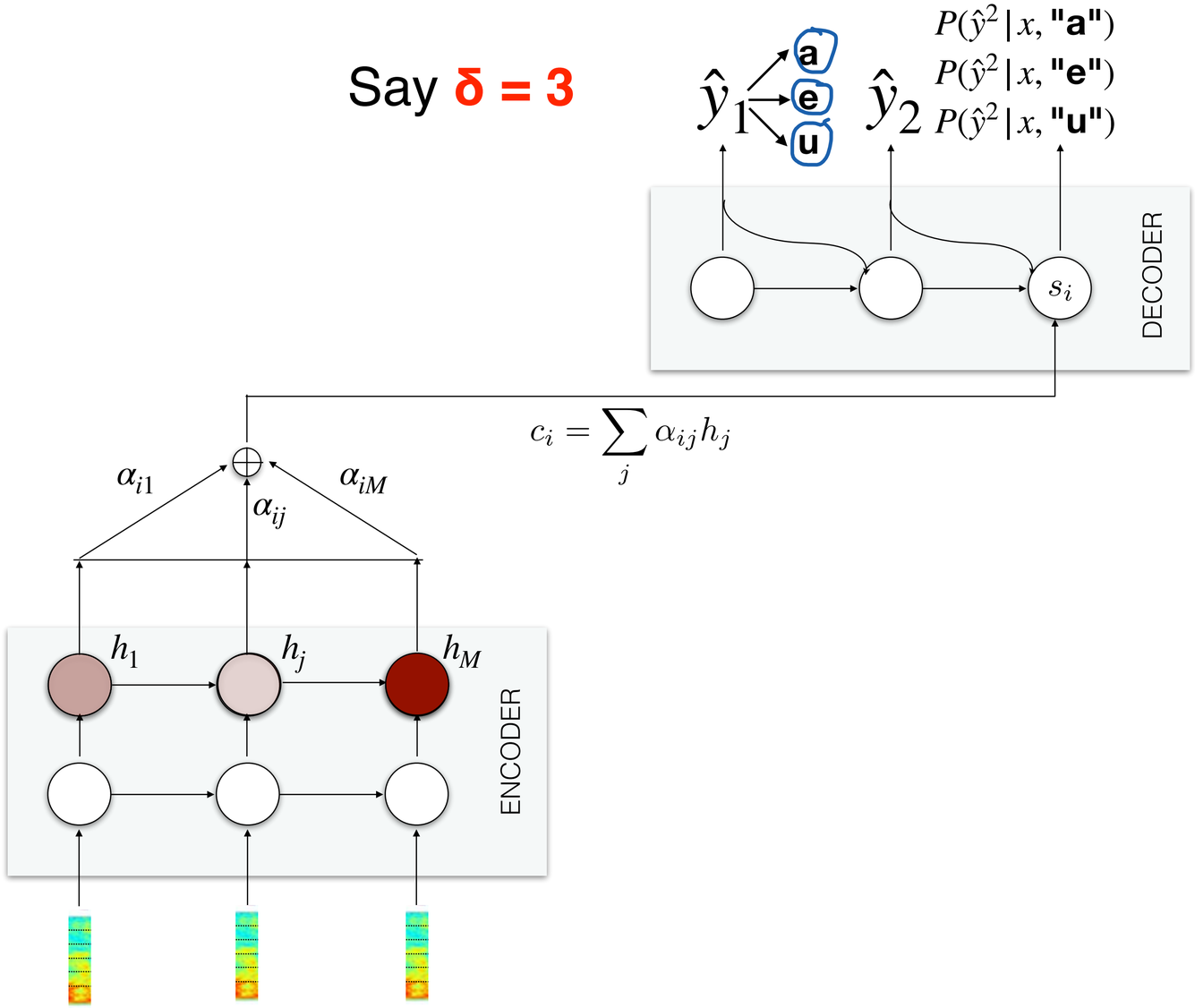
Beam search in a seq2seq model

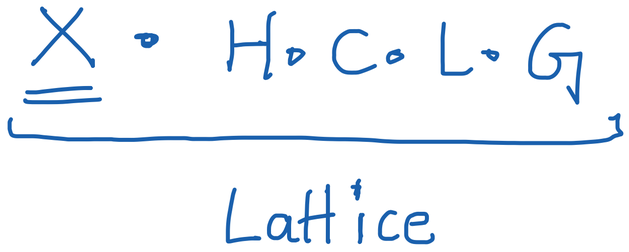
Say **δ = 3**



Beam search in a seq2seq model

Say **δ = 3**





$$w^* = \arg\max_w p(o|w)p(w)$$

Lattices

- "**Lattices**" are useful when more than one hypothesis is desired from a recognition pass

Lattices

- “**Lattices**” are useful when more than one hypothesis is desired from a recognition pass

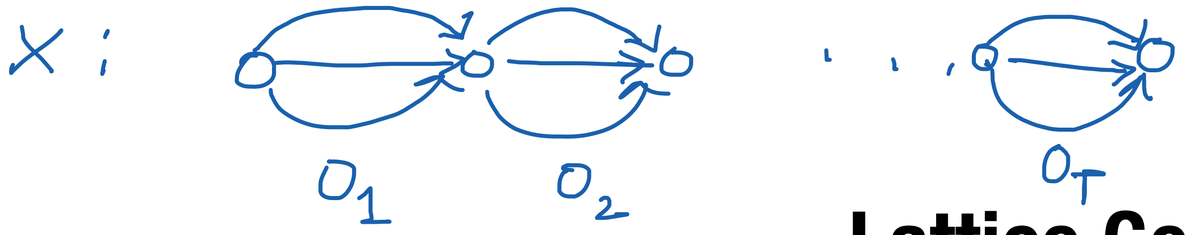
- A lattice is a weighted, directed acyclic graph which encodes a large number of ASR hypotheses weighted by acoustic model +language model scores specific to a given utterance

Lattice Generation

- Say we want to decode an utterance, U, of T frames.

Lattice Generation

- Say we want to decode an utterance, U, of T frames.
- Construct a sausage acceptor for this utterance, X, with T+1 states and arcs for each context-dependent HMM state at each time-step



Lattice Generation

- Say we want to decode an utterance, U, of T frames.
- Construct a sausage acceptor for this utterance, X, with T+1 states and arcs for each context-dependent HMM state at each time-step
- Search the following composed machine for the best word sequence corresponding to U:

$$D = X \circ HCLG$$

Lattice construction using lattice-beam

Lattice construction using lattice-beam

- To produce a lattice, prune the decoding graph using “lattice-beam” width (s.t. only arcs or states on paths that are within cutoff cost = best_path_cost + lattice-beam will be retained) and then determinize s.t. there’s a single path for every word sequence

Lattice construction using lattice-beam

- To produce a lattice, prune the decoding graph using “lattice-beam” width (s.t. only arcs or states on paths that are within cutoff cost = best_path_cost + lattice-beam will be retained) and then determinize s.t. there’s a single path for every word sequence
- Naive algorithm
  - Maintain a list of active tokens and links during decoding
  - Turn this structure into an FST, L.
  - When we reach the end of the utterance, prune L using lattice-beam.

Moving on to multi-pass decoding

- We learned about the beam search algorithm that helps search through the decoding graph in a first-pass decoding

Moving on to multi-pass decoding

- We learned about the beam search algorithm that helps search through the decoding graph in a first-pass decoding
- However, some models are too expensive to implement in first-pass decoding (e.g. RNN-based LMs)

Moving on to multi-pass decoding

- We learned about the beam search algorithm that helps search through the decoding graph in a first-pass decoding
- However, some models are too expensive to implement in first-pass decoding (e.g. RNN-based LMs)
- Multi-pass decoding:

Moving on to multi-pass decoding

- We learned about the beam search algorithm that helps search through the decoding graph in a first-pass decoding
- However, some models are too expensive to implement in first-pass decoding (e.g. RNN-based LMs)
- Multi-pass decoding:
  - First, use simpler model (e.g. Ngram LMs) to find most probable word sequences and represent as a word lattice or N-best list

Moving on to multi-pass decoding

- We learned about the beam search algorithm that helps search through the decoding graph in a first-pass decoding
- However, some models are too expensive to implement in first-pass decoding (e.g. RNN-based LMs)
- Multi-pass decoding:
  - First, use simpler model (e.g. Ngram LMs) to find most probable word sequences and represent as a word lattice or N-best list
  - Rescore first-pass hypotheses using complex model to find the best word sequence

Multi-pass decoding with N-best lists

- Simple algorithm: Modify the Viterbi algorithm to return the N-best word sequences for a given speech input

Rank	Path	AM logprob	LM logprob
1.	it's an area that's naturally sort of mysterious	-7193.53	-20.25
2.	that's an area that's naturally sort of mysterious	-7192.28	-21.11
3.	it's an area that's not really sort of mysterious	-7221.68	-18.91
4.	that scenario that's naturally sort of mysterious	-7189.19	-22.08
5.	there's an area that's naturally sort of mysterious	-7198.35	-21.34
6.	that's an area that's not really sort of mysterious	-7220.44	-19.77
7.	the scenario that's naturally sort of mysterious	-7205.42	-21.50
8.	so it's an area that's naturally sort of mysterious	-7195.92	-21.71
9.	that scenario that's not really sort of mysterious	-7217.34	-20.70
10.	there's an area that's not really sort of mysterious	-7226.51	-20.01

Image from [JM]: Jurafsky & Martin, SLP 2nd edition, Chapter 10

Multi-pass decoding with N-best lists

- Simple algorithm: Modify the Viterbi algorithm to return the N-best word sequences for a given speech input

Rank	Path	AM logprob	LM logprob
1.	it's an area that's naturally sort of mysterious	-7193.53	-20.25
2.	that's an area that's naturally sort of mysterious	-7192.28	-21.11
3.	it's an area that's not really sort of mysterious	-7221.68	-18.91
4.	that scenario that's naturally sort of mysterious	-7189.19	-22.08
5.	there's an area that's naturally sort of mysterious	-7198.35	-21.34
6.	that's an area that's not really sort of mysterious	-7220.44	-19.77
7.	the scenario that's naturally sort of mysterious	-7205.42	-21.50
8.	so it's an area that's naturally sort of mysterious	-7195.92	-21.71
9.	that scenario that's not really sort of mysterious	-7217.34	-20.70
10.	there's an area that's not really sort of mysterious	-7226.51	-20.01

- Problem: N-best lists aren't as diverse as we'd like. And, not enough information in N-best lists to effectively use other knowledge sources

Image from [JM]: Jurafsky & Martin, SLP 2nd edition, Chapter 10

Multi-pass decoding with lattices

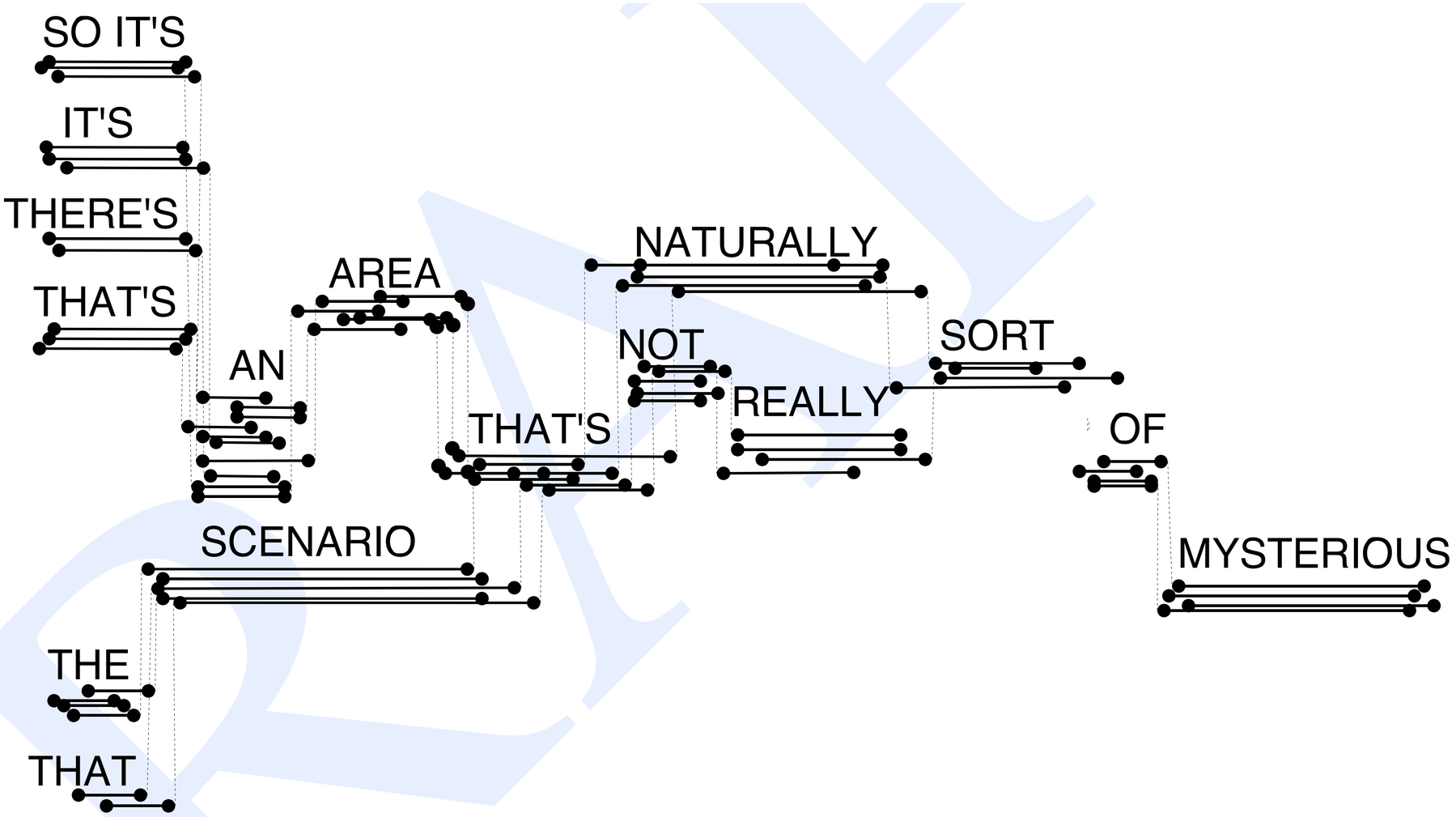


Image from [JM]: Jurafsky & Martin, SLP 2nd edition, Chapter 10