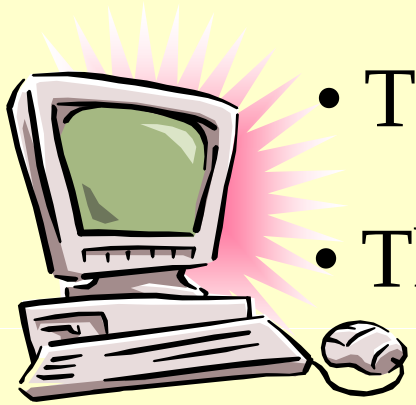
Speech Recognition (Language Modeling)

Sadaoki Furui
Tokyo Institute of Technology
Department of Computer Science
furui@cs.titech.ac.jp

Language model is crucial !

- Rudolph the red nose reindeer.
- Rudolph the Red knows rain, dear.
- Rudolph the Red Nose reigned here.

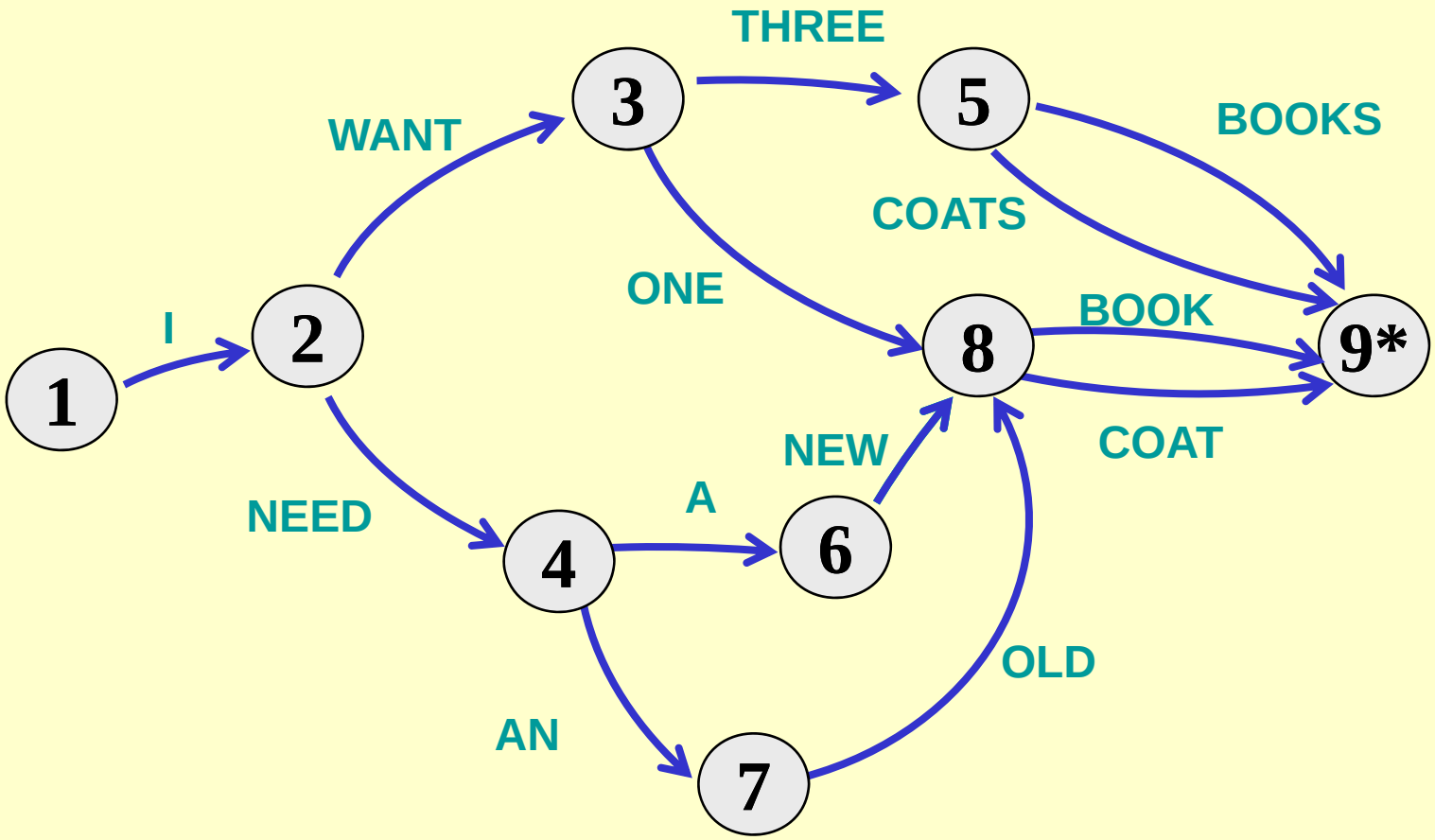


- This new display can recognize speech.
- This nudist play can wreck a nice beach.

Language model is crucial !

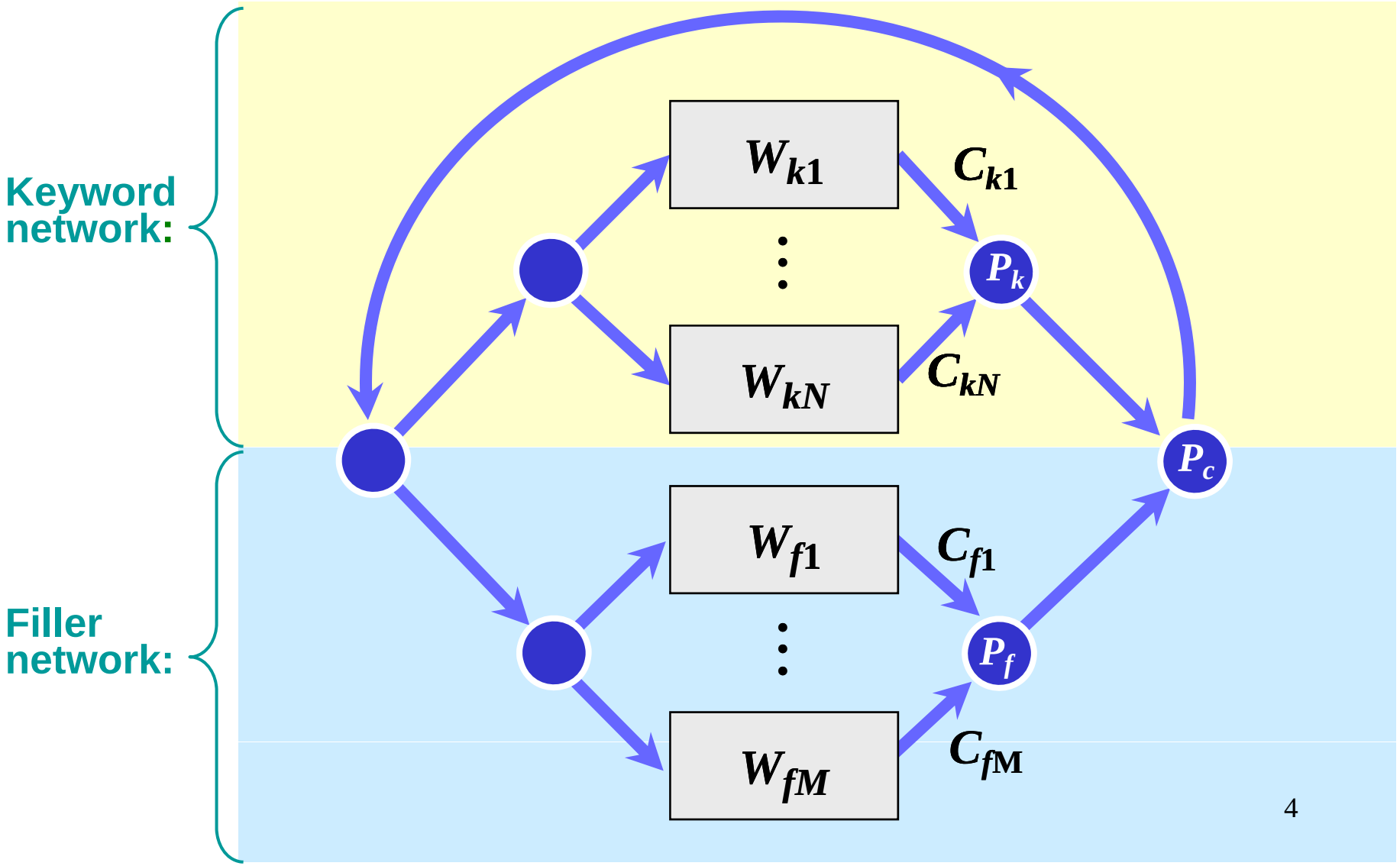
- It was a nice hockey game.
- It was an ice-hockey game.
- Did U2 have a nice trip?
- Did you two have a nice trip?

An example of FSN (Finite State Network) grammar

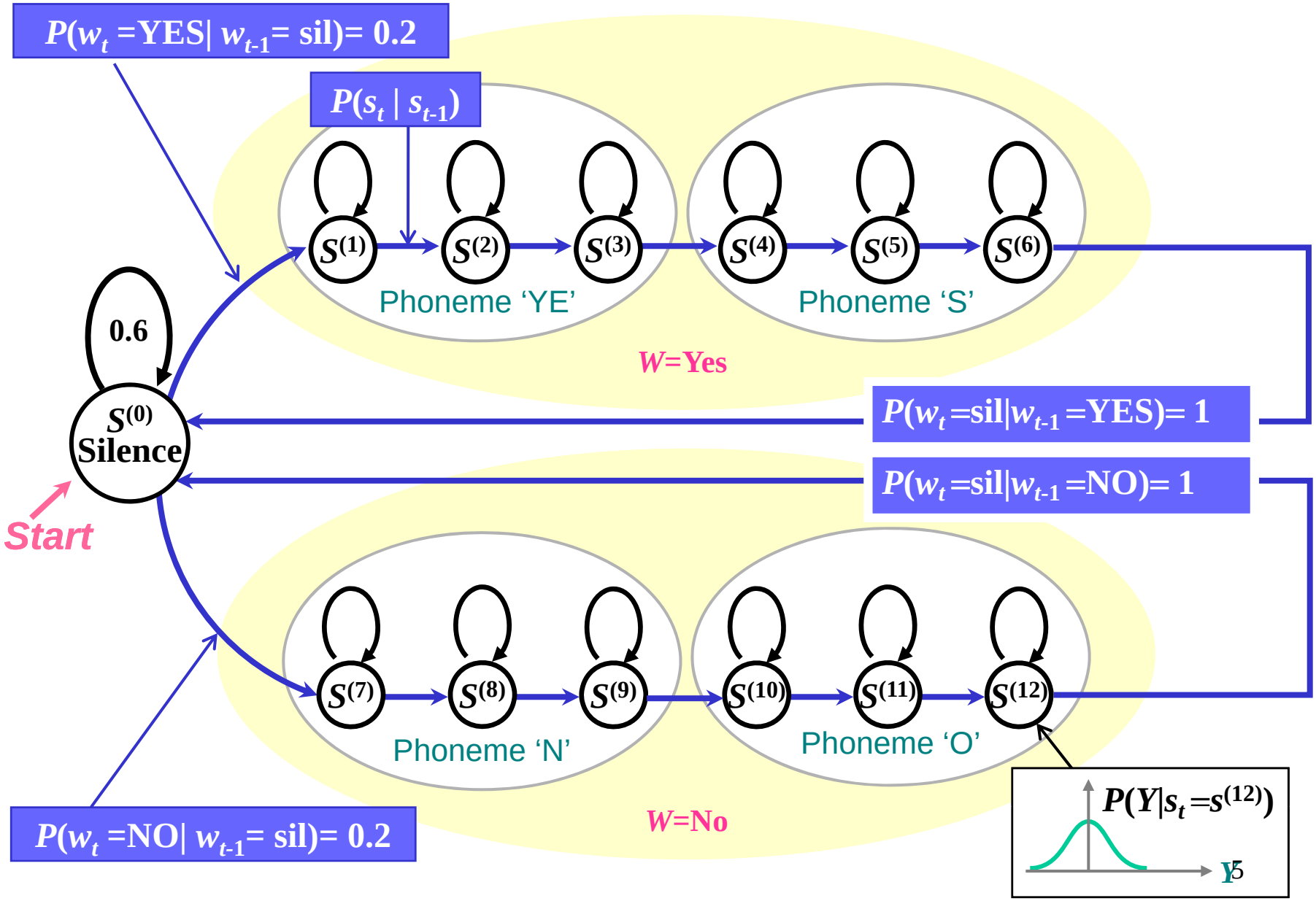


1.	I	5.	ONE	9.	BOOKS	13.	OLD
2.	WANT	6.	A	10.	COAT		
3.	NEED	7.	AN	11.	COATS		
4.	THREE	8.	BOOK	12.	NEW		

A simple, unconstrained finite state network containing N keywords W_{k1}, …, W_{kN} and M fillers W_{f1}, …, W_{fM}. Associated with each keyword and filler are word transition penalties C_{ki} and C_{fj} respectively.



Complete Hidden Markov Model of a simple grammar



Statistical language modeling

Probability of the word sequence $w_1^k = w_1 w_2 \ldots w_k$:

$$P(w_1^k) = \prod_{i=1}^{k} P(w_i \mid w_1 w_2 \ldots w_{i-1}) = \prod_{i=1}^{k} P(w_i \mid w_1^{i-1})$$

$$P(w_i \mid w_1^{i-1}) = N(w_1^i) \,/\, N(w_1^{i-1})$$

where $N(w_1^i)$ is the number of occurrences of the string w_1^i in the given training corpus.

Approximation by Markov processes:

Bigram model $\quad P(w_i \mid w_1^{i-1}) = P(w_i \mid w_{i-1})$

Trigram model $\quad P(w_i \mid w_1^{i-1}) = P(w_i \mid w_{i-2} w_{i-1})$

Smoothing of trigram by the deleted interpolation method:

$$P(w_i \mid w_{i-2} w_{i-1}) = \lambda_1 P(w_i \mid w_{i-2} w_{i-1}) + \lambda_2 P(w_i \mid w_{i-1}) + \lambda_3 P(w_i)$$

Good-Turing estimate

For any *n*-gram that occurs *r* times, we should pretend that it occurs *r times as follows:**

$$r^* = (r+1)\frac{n_{r+1}}{n_r}$$

where n_r is the number of *n*-grams that occur exactly *r* times in the training data.

Katz smoothing algorithm

Katz smoothing extends the intuitions of the Good-Turing estimate by adding the combination of higher-order models with lower-order models.

$$P_{Kats}(w_i|w_{i\text{-}1}) = \begin{cases} C(w_{i-1}w_i)/C(w_{i-1}) & \text{if } r>k \\ d_r C(w_{i-1}w_i)/C(w_{i-1}) & \text{if } k \geq r > 0 \\ \alpha(w_{i-1})P(w_i) & \text{if } r = 0 \end{cases}$$

where $d_r = \dfrac{\dfrac{r^*}{r} - \dfrac{(k+1)\,n_{k+1}}{n_1}}{1 - \dfrac{(k+1)\,n_{k+1}}{n_1}}$ and $\alpha(w_{i-1}) = \dfrac{1 - \sum_{w_i : r>0} P_{Kats}(w_i|w_{i\text{-}1})}{1 - \sum_{w_i : r>0} P(w_i)}$

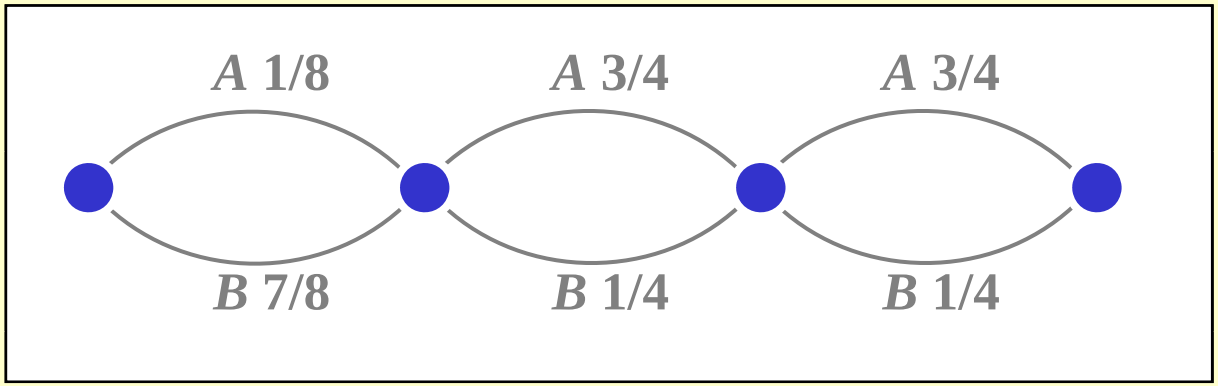
Task difficulty (Entropy)

- **Yes, No** — **1 bit (- log 0.5)**
- **Digits**
  - * **0 ~ 9 : 0.1** — **3.32 bits (- log 0.1)**
  - * **0 : 0.91**, **1 ~ 9 : 0.01** — **0.722 bits**

    (- 0.91 log 0.91 - 0.09 log 0.01)

Test-set perplexity

Vocabulary: *A*, *B* Test sentence: *ABB*

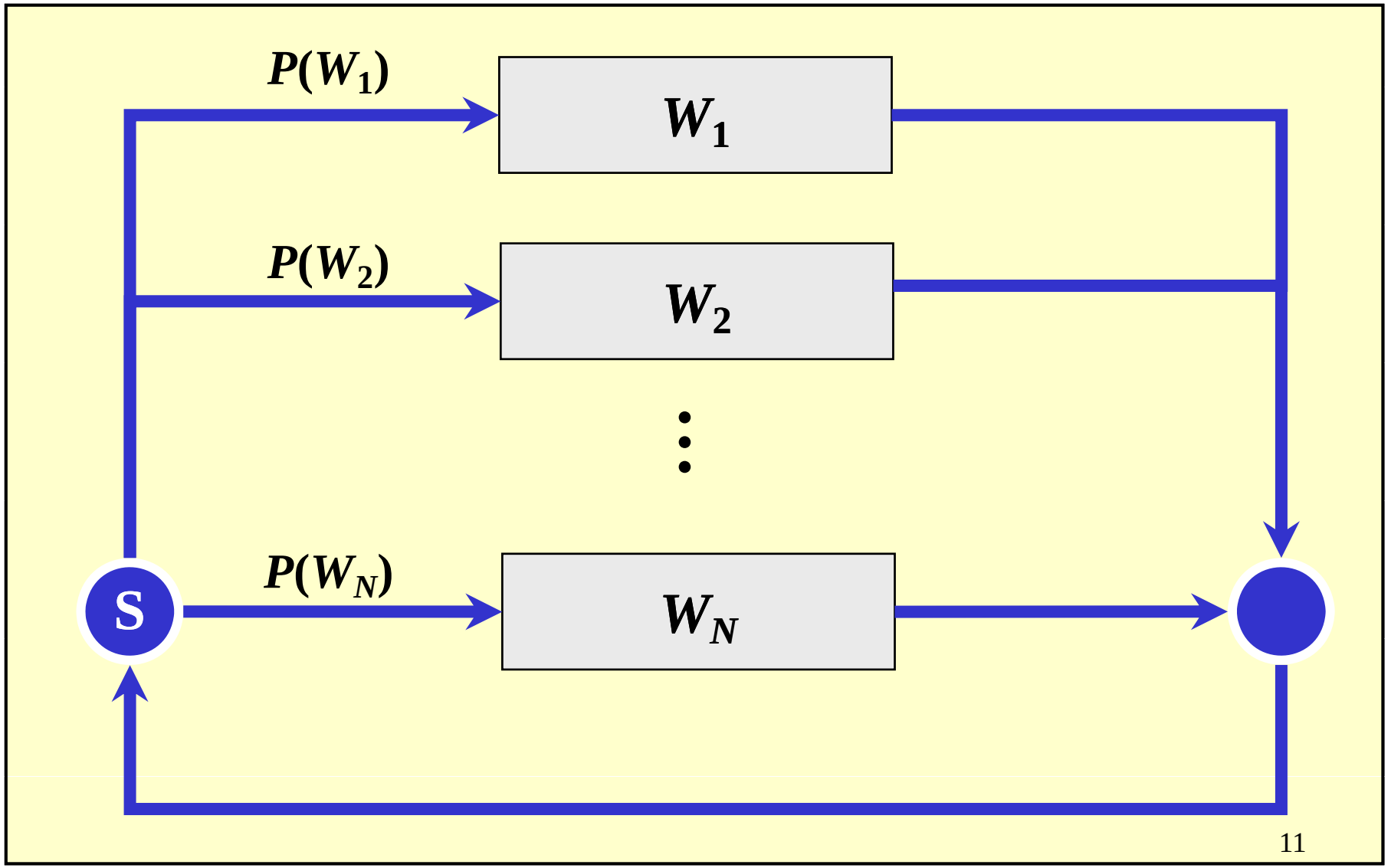


Test sentence entropy: -log 1/8 - log 1/4 - log 1/4 = 7 bits
(*ABB*)

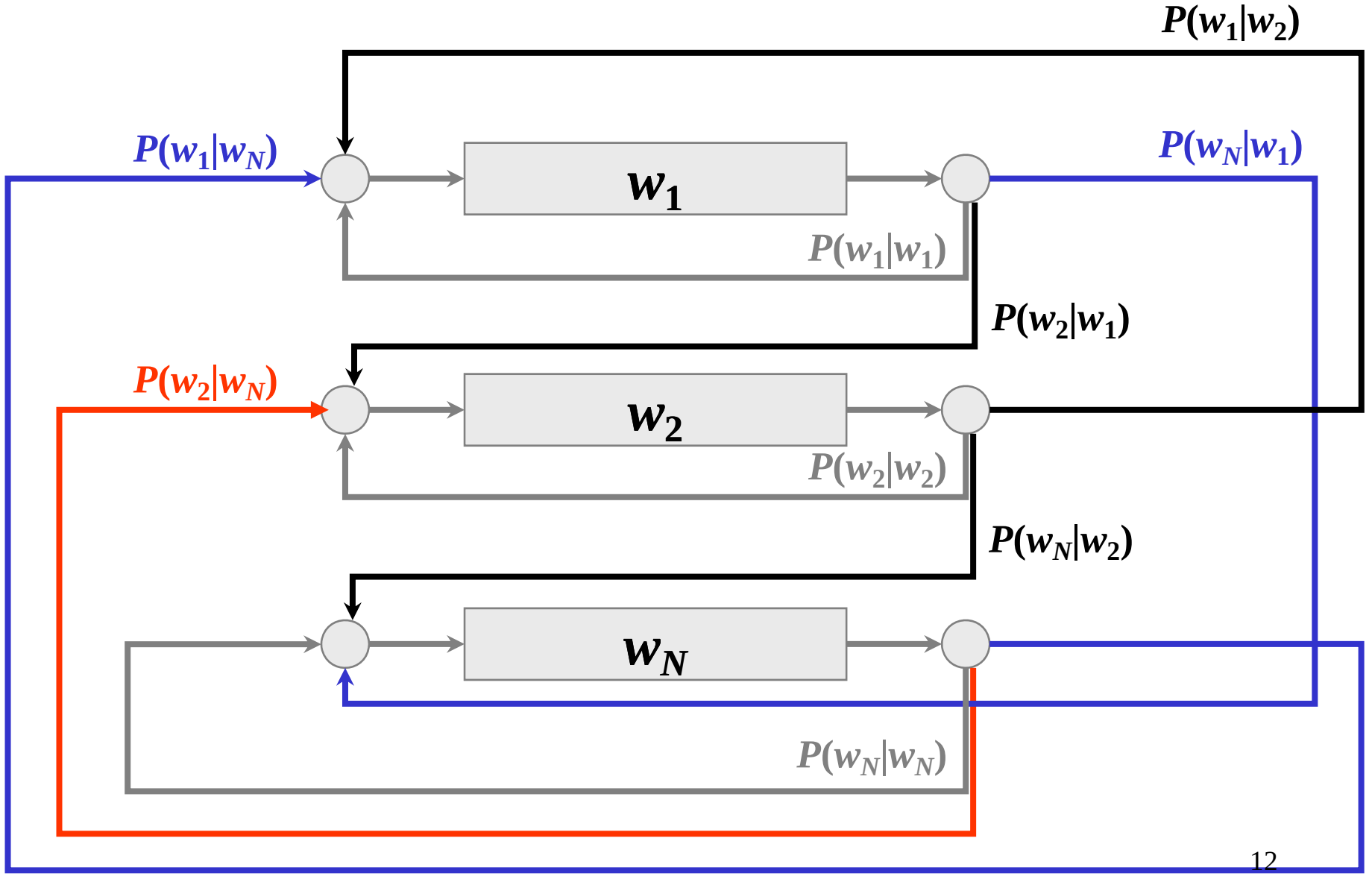
Entropy per word: 7 / 3 = 2.33 bits

Test-set perplexity: $2^{2.33}$ = 5.01

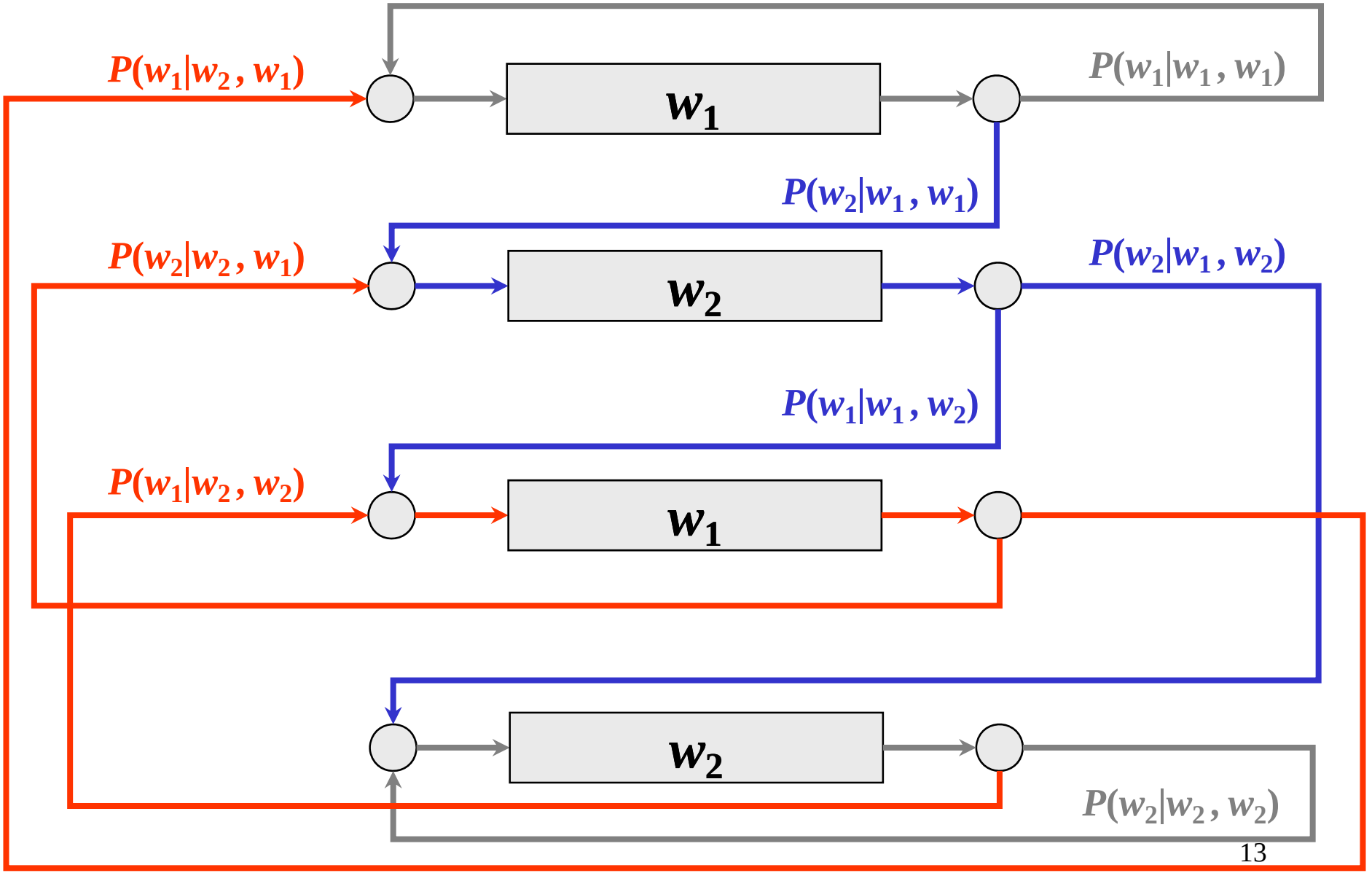
A unigram grammar network where the unigram probability is attached as the transition probability from starting state S to the first state of each word HMM.



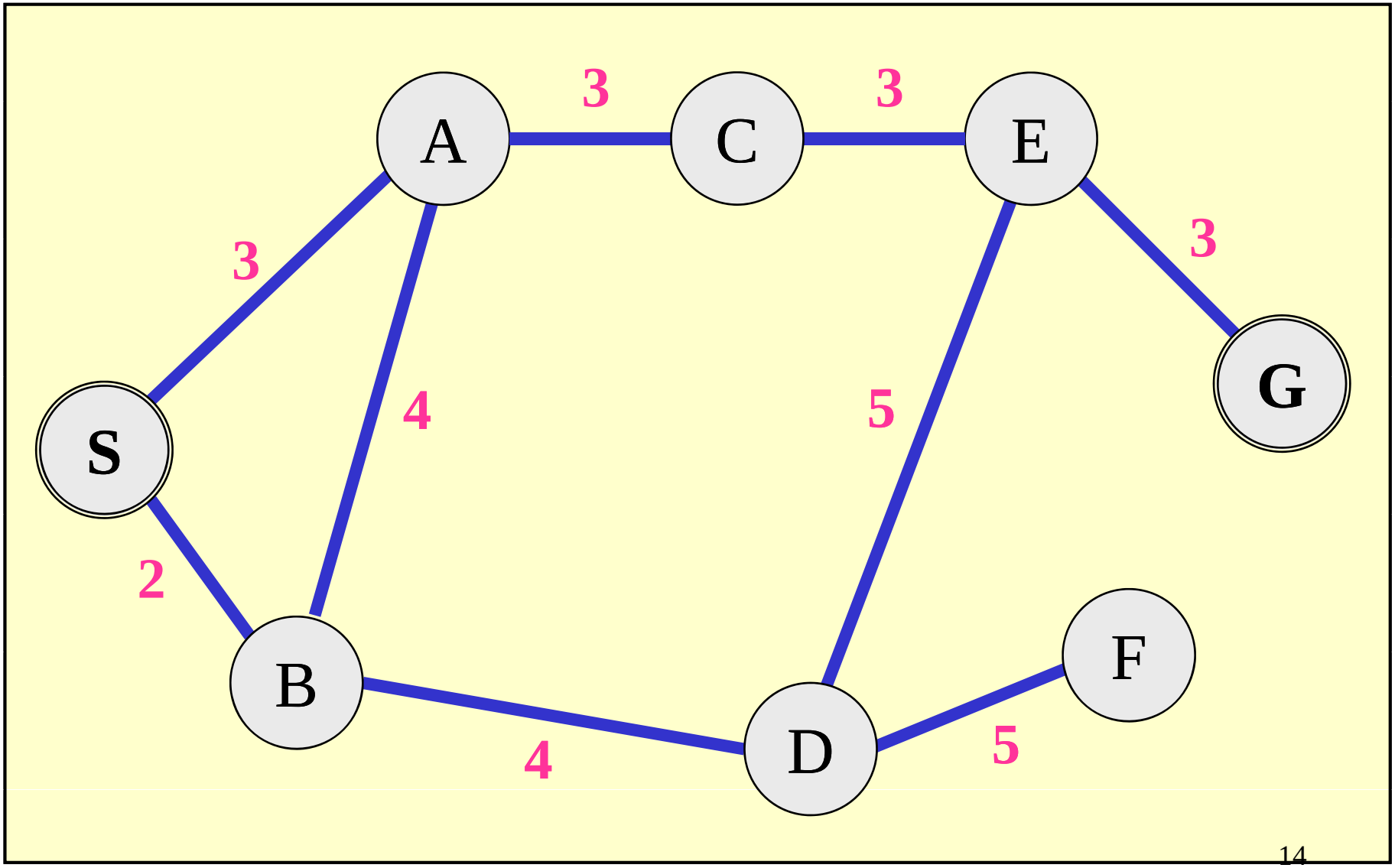
A bigram grammar network where the bigram probability $P(w_j|w_i)$ is attached as the transition probability from word w_i to w_j.



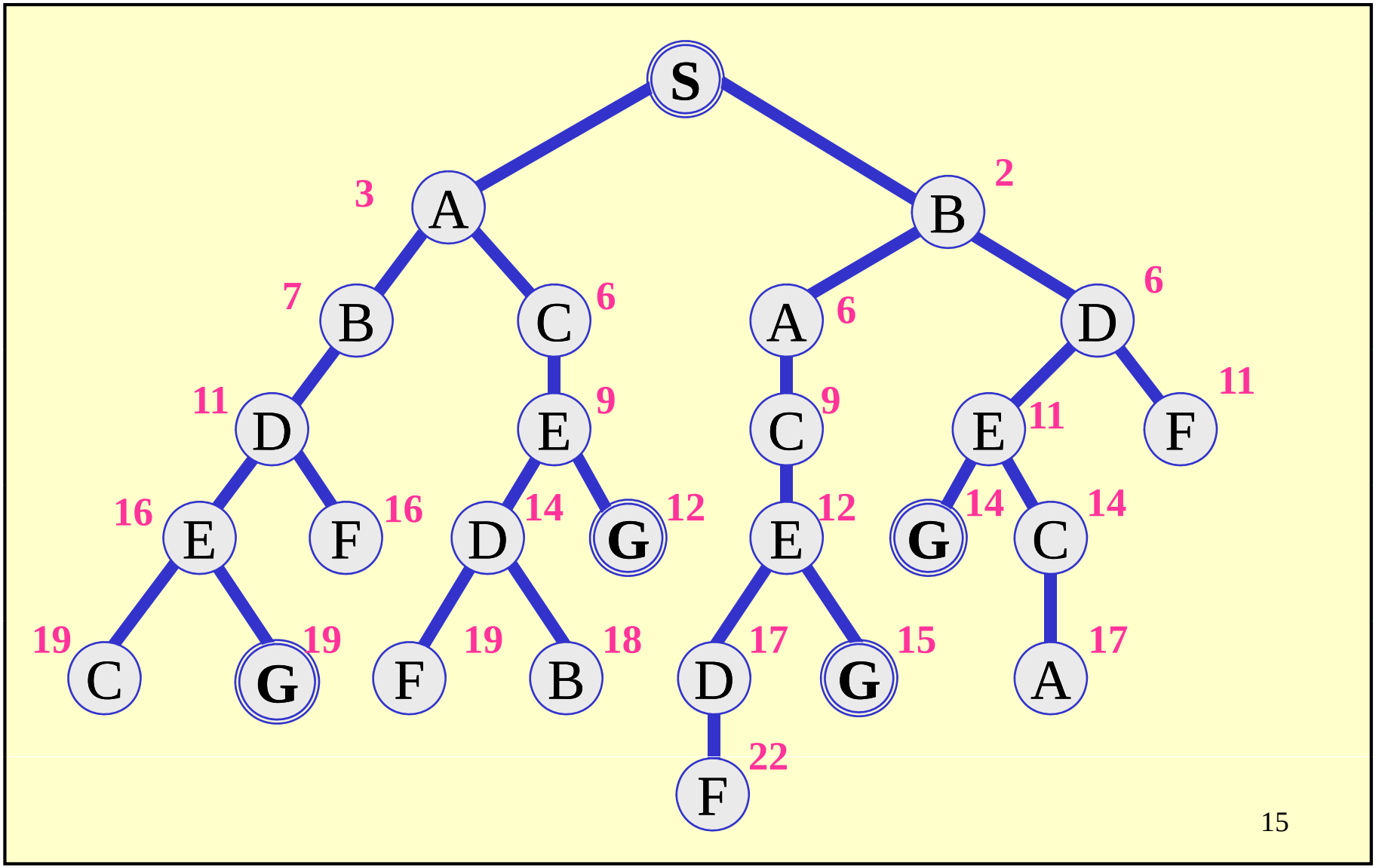
A trigram grammar network where the trigram probability $P(w_k|w_i, w_j)$ is attached to transition from grammar state w_i, w_j to the next word w_k. Illustrated here is a two-word vocabulary, so there are four grammar states in the trigram network.



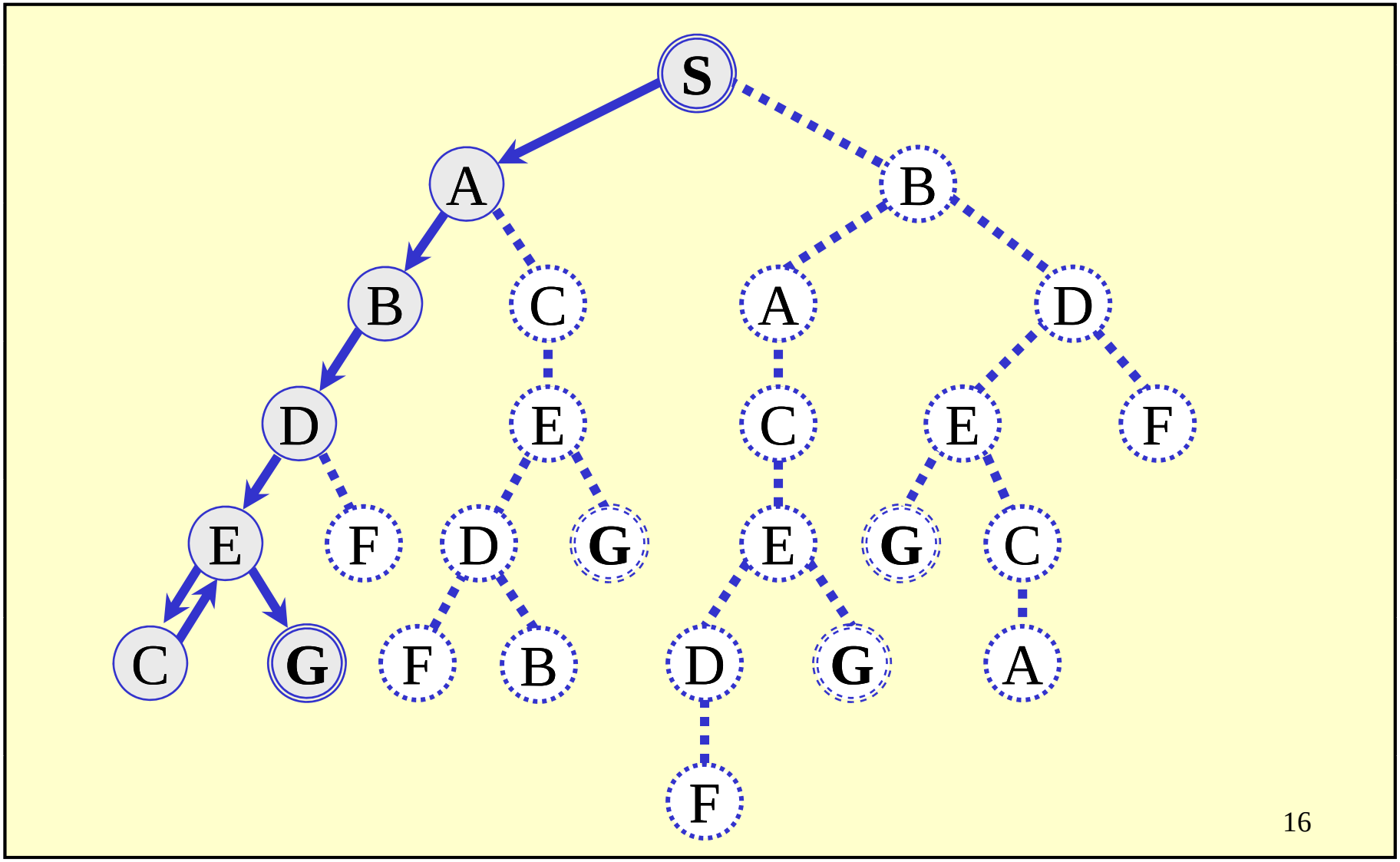
A highway distance map for cities S, A, B, C, D, E, F, and G.
The salesman needs to find a path to travel from city S to city G.



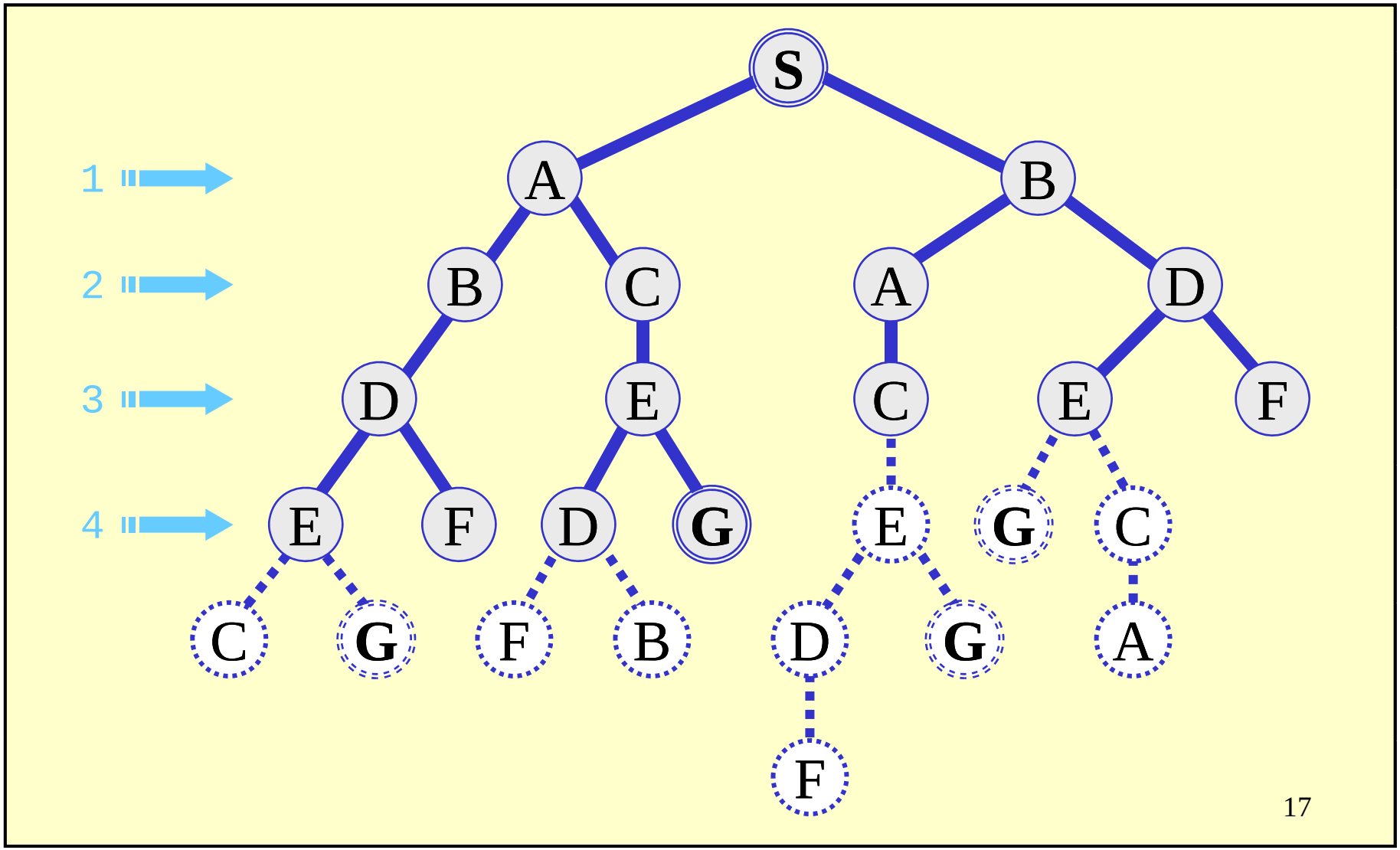
The search tree (graph) for the salesman problem illustrated in the previous figure. The number next to each node is the accumulated distance from start city to end city.



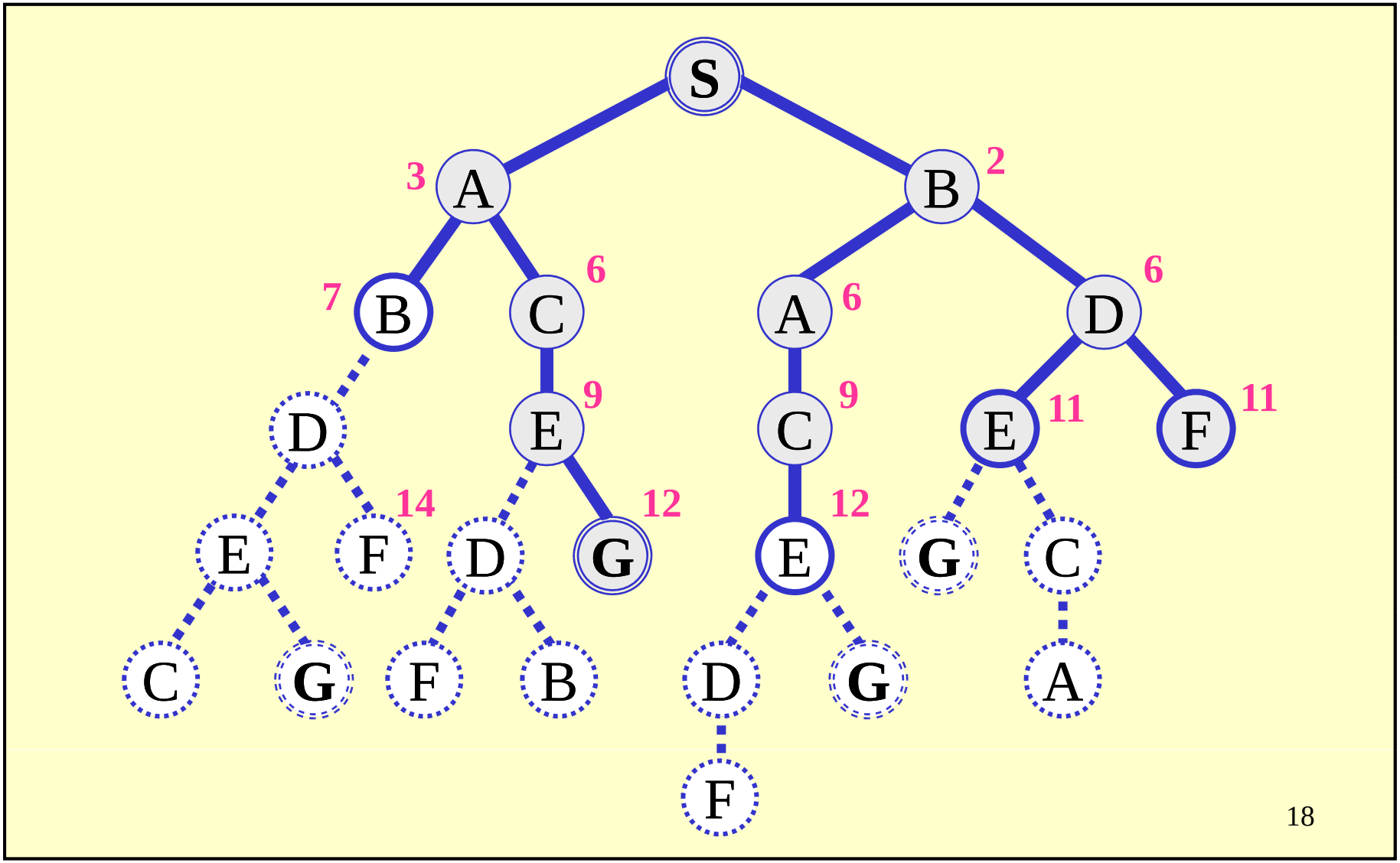
The node-expanding procedure of the depth-first search for the path search problem. When it fails to find the goal city in node C, it backtracks to the parent and continues the search until it finds the goal city. The green nodes are those that are explored. The doted nodes are not visited during the search.



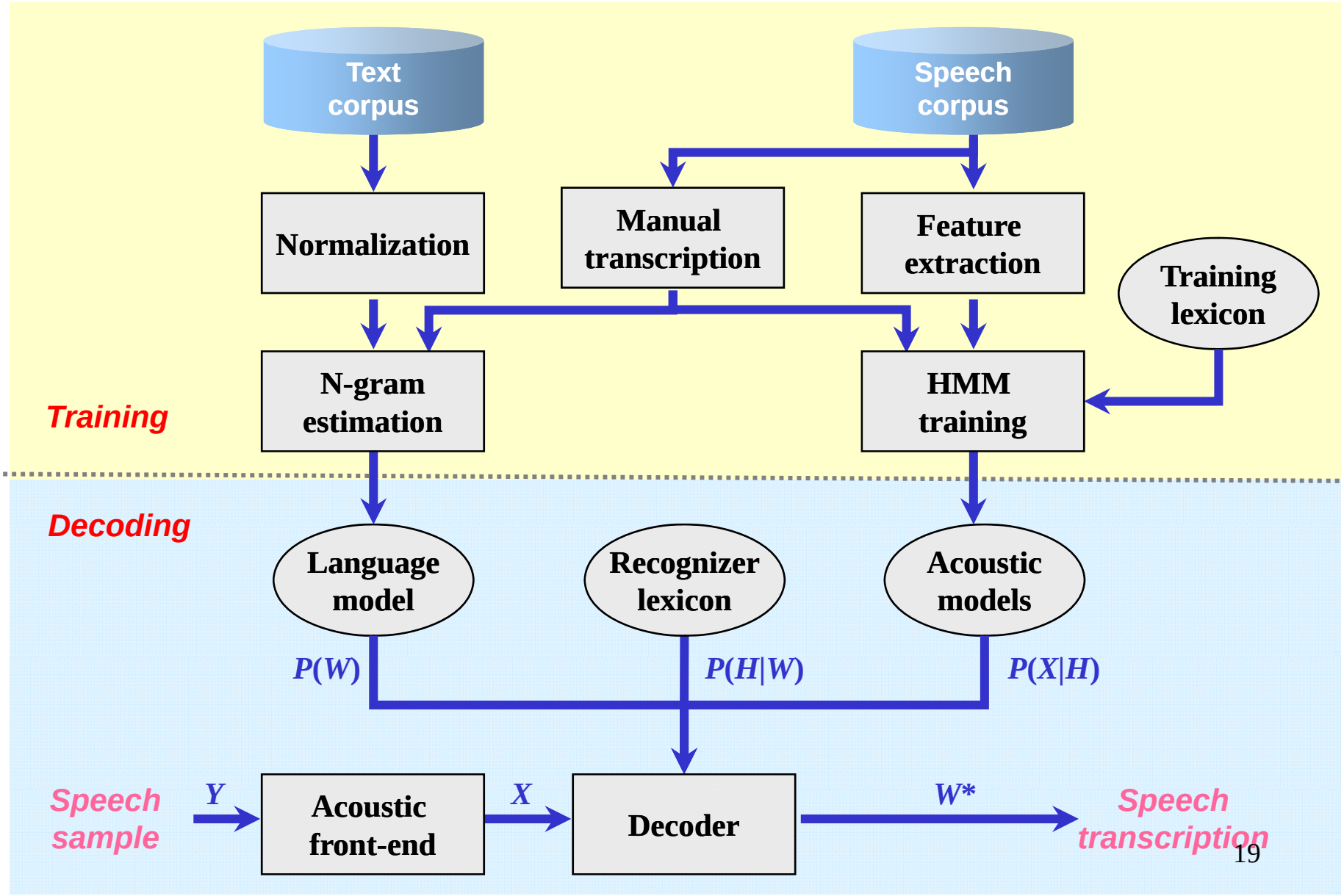
The node-expanding procedure of a breath-first search for the path search problem. It searches through each level until the goal is identified. The green nodes are those that are explored. The dotted nodes are not visited during the search.



Beam search for the city-travel problem. The nodes with green color are the ones kept in the beam. The transparent nodes were explored but pruned because of higher cost. The dotted nodes indicate all the savings because of pruning.



System diagram of a generic speech recognizer based on statistical models, including training and decoding processes and the main knowledge sources.



Example word lattice generated by a speech recognizer using a bigram language model for a 2.1s utterance. Each graph edge corresponds to a word hypothesis and a time interval (as specified by the time information on the nodes).