Hidden Markov Model (HMM)

HMM

- **The HMM is a probabilistic sequence model.**
- A sequence model/classifier is a model whose job is to assign a label/class to each unit in a sequence, thus mapping a sequence of observations to a sequence of labels.
- Given a sequence of units (words, letters, morphemes, sentences, whatever), it computes a probability distribution over possible sequences of labels and choose the best label sequence.
- It is one of the **most important machine learning models in speech and language processing**

- Markov chain is only useful for assigning probabilities to unambiguous sequences.

- A Markov chain is specified by the following components:

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{01} a_{02} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
q_0, q_F	a special **start state** and **end (final) state** that are not associated with observations

Markov Chain: "First-order observable Markov Model"

- The probability of a particular state depends only on the previous state:

$$P(q_i \mid q_1 \ldots q_{i-1}) = P(q_i \mid q_{i-1})$$

$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$
$QA = \{q_x, q_y \ldots\}$	a set $QA \subset Q$ of legal **accepting states**

Example

According to Kemeny, Snell, and Thompson, the Land of island XYZ is blessed by many things, but not by good weather.

They never have two nice days in a row.

If they have a nice day, they are just as likely to have snow as rain the next day.

If they have snow or rain, they have an even chance of having the same the next day.

If there is change from snow or rain, only half of the time is this a change to a nice day. With this information we form a Markov chain as follows.

We take as states the kinds of weather R, N, and S. From the above information we determine the transition probabilities. Write the transition probability Matrix A

$$\begin{array}{c} \\ \mathrm{R} \\ \mathrm{N} \\ \mathrm{S} \end{array} \begin{array}{c} \begin{array}{ccc} \mathrm{R} & \mathrm{N} & \mathrm{S} \end{array} \\ \begin{pmatrix} 1/2 & 1/4 & 1/4 \\ 1/2 & 0 & 1/2 \\ 1/4 & 1/4 & 1/2 \end{pmatrix} \end{array}$$

The Hidden Markov Model

- A Markov chain is useful when we need to compute a probability for a sequence of events that we can observe in the world.
- It allows us to talk about **both observed events** (**like words** that we see in the input) **and hidden events (like part-of-speech tags)** that we think of as causal factors in our probabilistic model.

Example HMM

- Imagine that **you** are a climatologist in the year 2799 studying the history of global warming. You cannot find any records of the weather in Baltimore, Maryland, for the summer of 2007
- But you do find **Jason Eisner's diary**, which lists how many ice creams Jason ate every day that summer.
- Our goal is to use these observations to estimate the temperature every day.
- Assume two kinds of days: cold (C) and hot (H).
- So the **Eisner task is** : *Given a sequence of observations O, each observation an integer corresponding to the number of ice creams eaten on a given day, figure out the correct 'hidden' sequence Q of weather states (H or C) which caused Jason to eat the ice cream*.

HMM components

- An HMM is specified by the following components

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
q_0, q_F	a special **start state** and **end (final) state** that are not associated with observations, together with transition probabilities $a_{01} a_{02} \ldots a_{0n}$ out of the start state and $a_{1F} a_{2F} \ldots a_{nF}$ into the end state

First-order HMM

A first-order hidden Markov model instantiates two simplifying assumptions.

- First, as with a first-order Markov chain, **the probability of a particular state depends only on the previous state**:

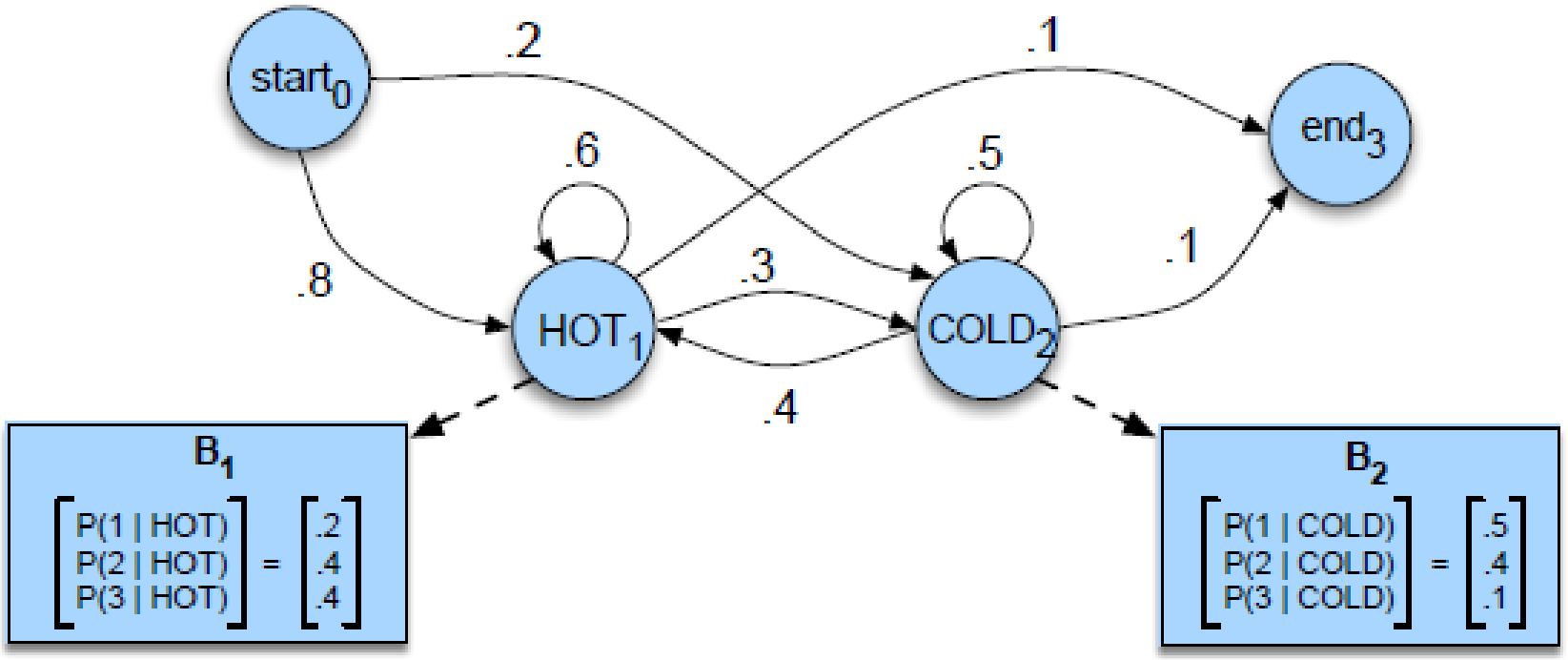
  Markov Assumption: $P(qi \mid q_1 ... q_{i-1}) = P(q_i \mid q_{i-1})$

- Second, the **probability of an output observation o_i depends only on the state that produced the observation q_i** and **not on any other states or any other observations:**

  Output Independence: $P(o_i|q_1 ... q_i ... q_T ; o_1 ... o_i ... o_T) = P(o_i|q_i)$

Example

- A sample HMM for the ice cream task. The two hidden states (H and C) correspond to hot and cold weather, and the observations (O = 1,2,3) correspond to the number of ice creams eaten by Jason on a given day.



HMM should be characterized by three fundamental problems

- **Problem 1 (Likelihood):**

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

- **Problem 2 (Decoding):**

  Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.

- **Problem 3 (Learning):**

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

HMM should be characterized by three fundamental problems

- **Problem 1 (Likelihood):**
  Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

- **Problem 2 (Decoding):**
  Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.

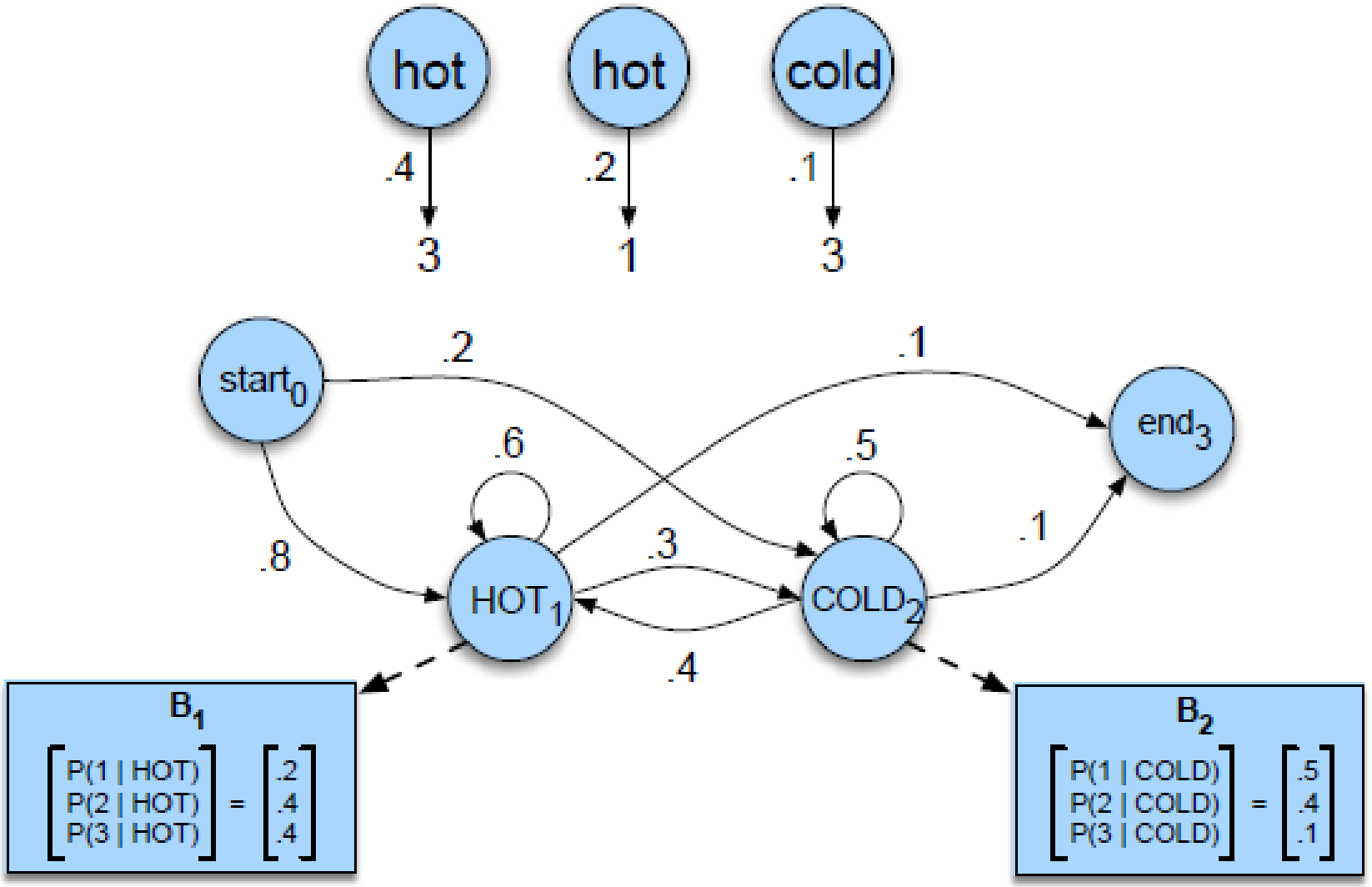
- **Problem 3 (Learning):**
  Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

Likelihood Computation: The Forward Algorithm

- Given an HMM λ = (A,B) and an observation sequence O, determine the likelihood P(O| λ).
- The likelihood of the observation sequence is

$$P(O|Q) = \prod_{i=1}^{T} P(o_i|q_i)$$

- **Observation 3 1 3**
- P(3 1 3| hot hot cold) = P(3|hot)P(1|hot)P(3|cold)



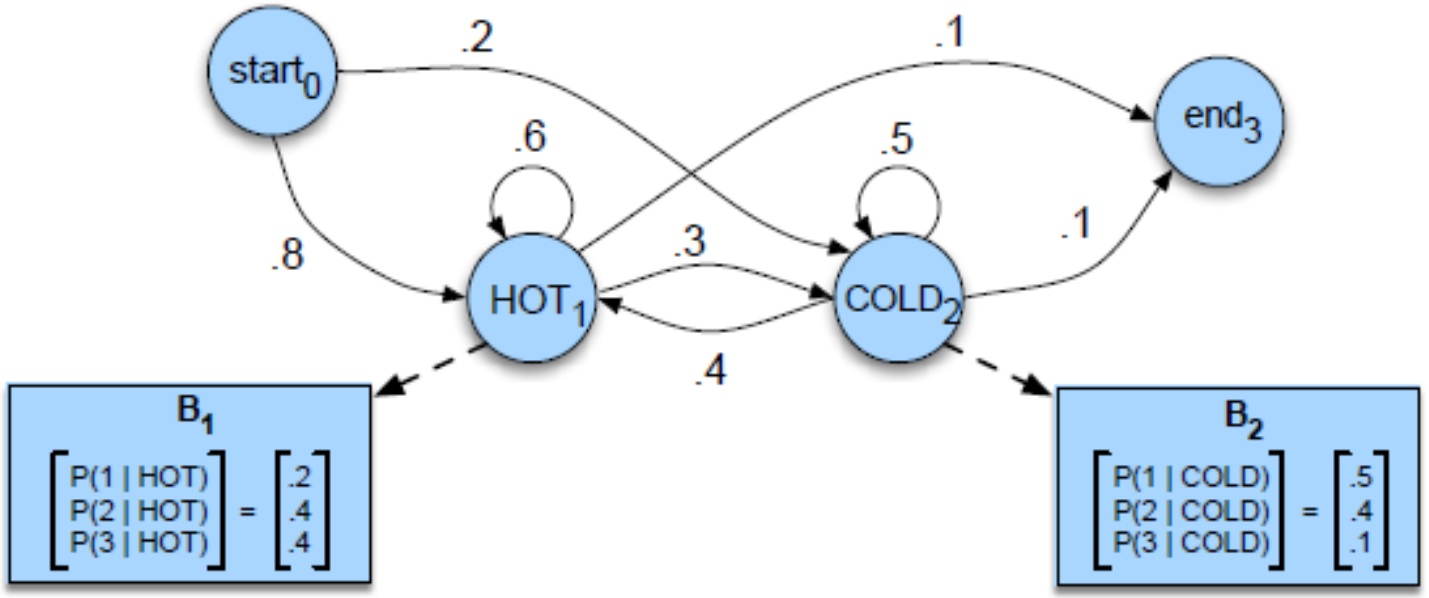
Joint probability

- Joint probability of being in a particular weather sequence Q and generating a *particular sequence O of ice-cream events*

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{T} P(o_i|q_i) \times \prod_{i=1}^{T} P(q_i|q_{i-1})$$

- ice-cream observation **3 1 3** and
- one possible hidden state sequence **hot hot cold**

$$P(3\ 1\ 3, \text{hot hot cold}) = P(\text{hot}|\text{start}) \times P(\text{hot}|\text{hot}) \times P(\text{cold}|\text{hot}) \times P(3|\text{hot}) \times P(1|\text{hot}) \times P(3|\text{cold})$$

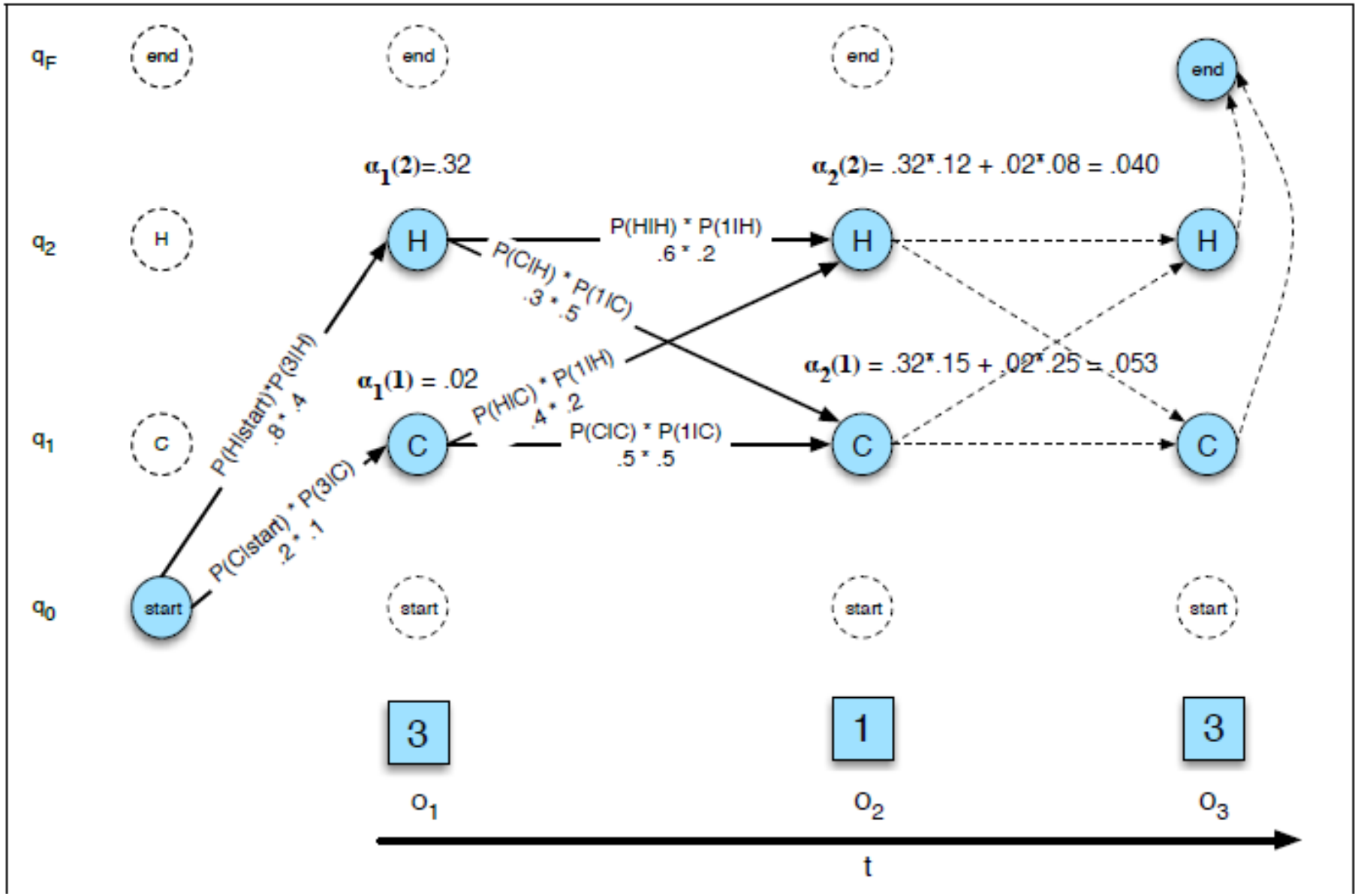


- The total probability of the observations just by summing over all possible hidden state sequences

$$P(O) = \sum_{Q} P(O,Q) = \sum_{Q} P(O|Q)P(Q)$$

- For an HMM with **N hidden states** and an observation sequence of T observations, there are **N^T possible hidden sequences.**
- For real tasks, where N and T are both large, **N^T is a very large number**, so we cannot compute the total observation likelihood by computing a separate observation likelihood for each hidden state sequence and then summing them.
- Instead of using such an extremely exponential algorithm, we use an efficient **Forward $O(N^2T)$ algorithm called the forward algorithm**

forward trellis for computing the likelihood of **3 1 3** given the hidden state sequence **hot hot cold**



- Each cell of the forward algorithm trellis $\alpha_t(j)$ represents the probability of being in state j after seeing the first t observations, given the automaton λ.
- The value of each cell $\alpha_t(j)$ is computed by summing over the probabilities of every path that could lead us to this cell.

$$\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

- $q_t = j$ means "the t *th* state in the sequence of states is state j".

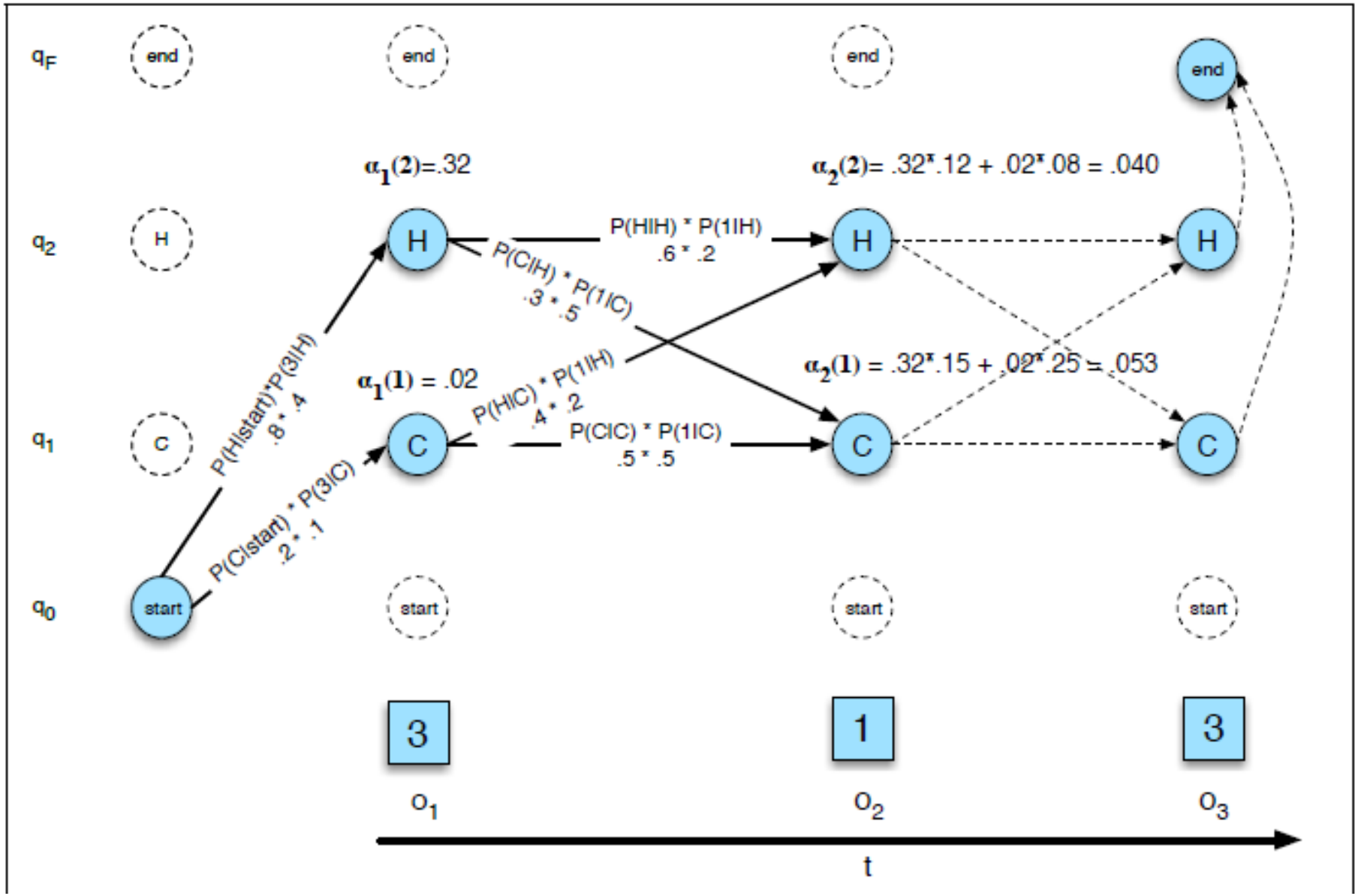
- $\alpha_t(j)$ is computed as

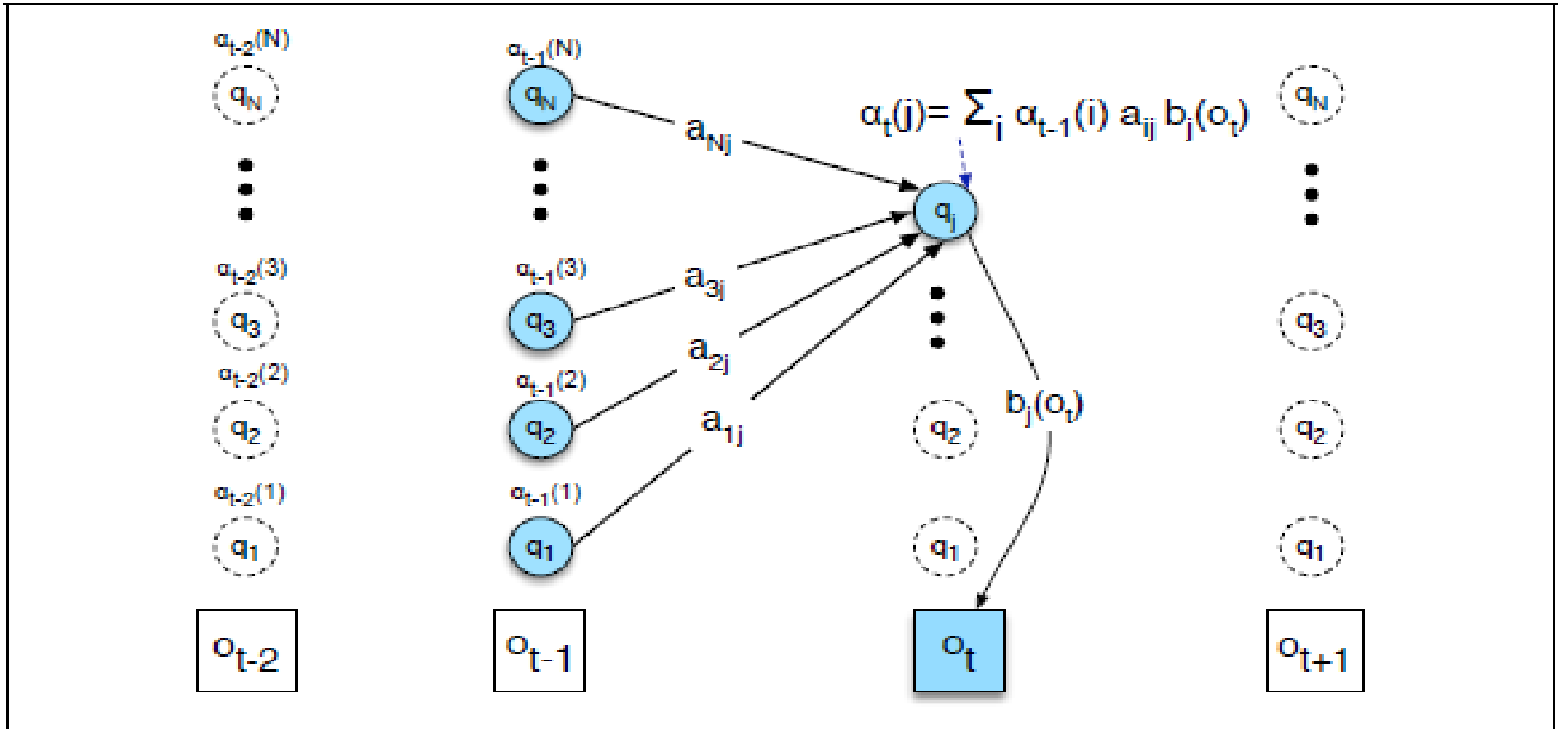
$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$$

- The three factors that are multiplied in extending the previous paths to compute the forward probability at time t are

$\alpha_{t-1}(i)$	the **previous forward path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

forward trellis for computing the likelihood of **3 1 3** given the hidden state sequence **hot hot cold**




$\alpha_{t-2}(N)$
q_N
$\alpha_{t-2}(3)$
q_3
$\alpha_{t-2}(2)$
q_2
$\alpha_{t-2}(1)$
q_1
o_{t-2}
$\alpha_{t-1}(N)$
q_N
$\alpha_{t-1}(3)$
q_3
$\alpha_{t-1}(2)$
q_2
$\alpha_{t-1}(1)$
q_1
o_{t-1}
a_{Nj}
a_{3j}
a_{2j}
a_{1j}
$\alpha_t(j) = \sum_i \alpha_{t-1}(i)\, a_{ij}\, b_j(o_t)$
q_j
q_N
q_3
q_2
q_1
$b_j(o_t)$
o_t
q_N
q_3
q_2
q_1
o_{t+1}

HMM should be characterized by three fundamental problems

- **Problem 1 (Likelihood):**
  Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

- **Problem 2 (Decoding):**
  Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.

- **Problem 3 (Learning):**
  Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

Decoding: The Viterbi Algorithm

- For any model, such as an HMM, that contains hidden variables**, the task of determining which sequence of variables is the underlying source of some sequence of observations is called the decoding task**.
- In the ice-cream domain, given a sequence of ice-cream observations 3 1 3 and an HMM, the task of the decoder is to **find the best hidden weather sequence (H H H).**

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

- The value of each cell $v_t(j)$ is computed by recursively taking the most probable path that could lead us to this cell.
- Each cell expresses the probability

$$v_t(j) = \max_{q_0, q_1, \ldots, q_{t-1}} P(q_0, q_1 \ldots q_{t-1}, o_1, o_2 \ldots o_t, q_t = j | \lambda)$$

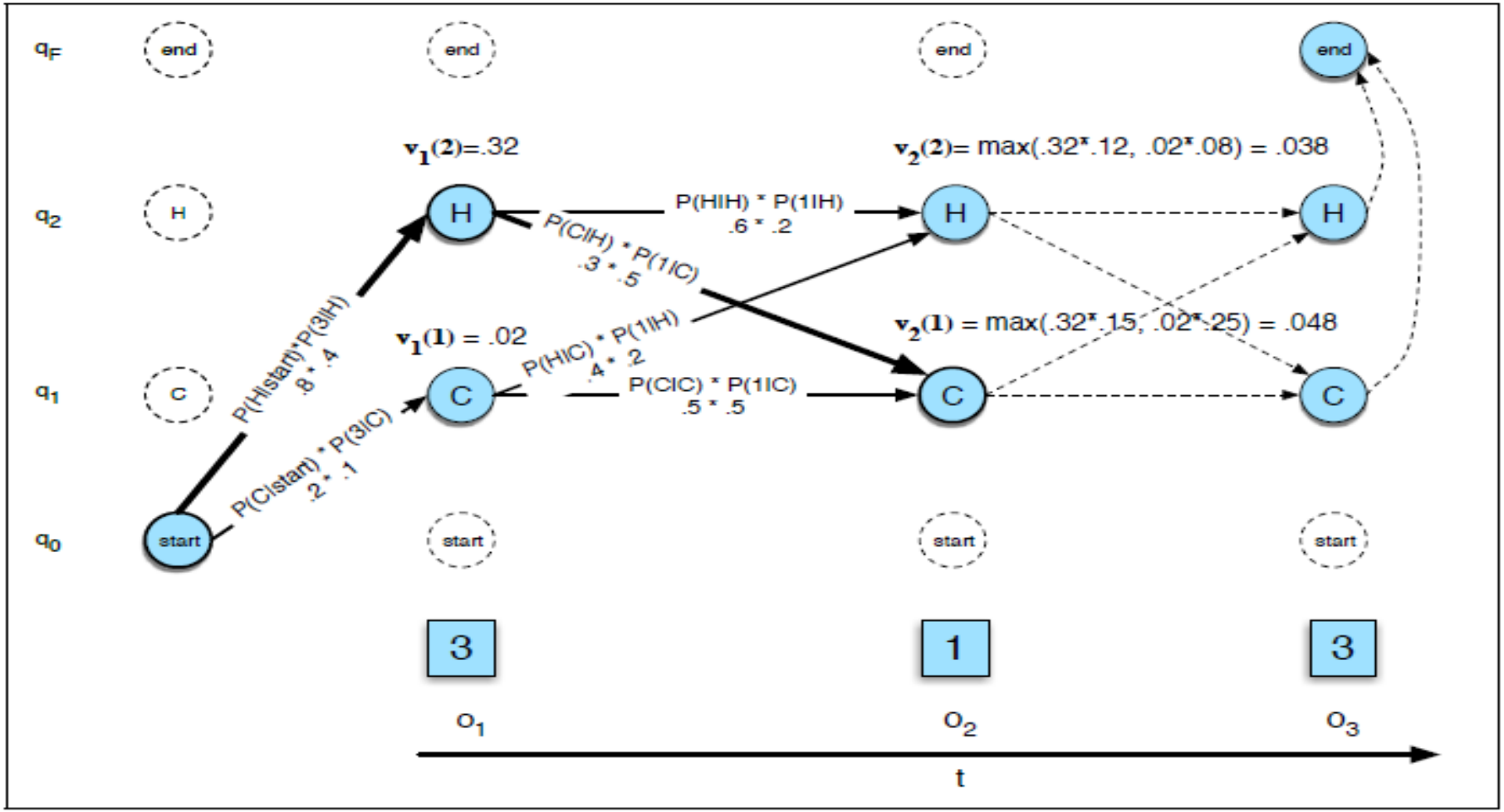
- For a given state q_j at time t, the value $v_t(j)$ is computed as

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

- The three factors that are multiplied for extending the previous paths to compute the Viterbi probability at time t are

$v_{t-1}(i)$	the **previous Viterbi path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

Viterbi trellis for computing the best hidden state sequence for the observation sequence 3 1 3.



HMM should be characterized by three fundamental problems

- **Problem 1 (Likelihood):**
  Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

- **Problem 2 (Decoding):**
  Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.

- **Problem 3 (Learning):**
  Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.