SciencePG
Science Publishing Group

Methodology Article

Longest-path Algorithm to Solve Uncovering Problem of Hidden Markov Model

Loc Nguyen

Sunflower Soft Company, Ho Chi Minh City, Vietnam

Email address:

ng_phloc@yahoo.com

To cite this article:

Abstract: Uncovering problem is one of three main problems of hidden Markov model (HMM), which aims to find out optimal state sequence that is most likely to produce a given observation sequence. Although Viterbi is the best algorithm to solve uncovering problem, I introduce a new viewpoint of how to solve HMM uncovering problem. The proposed algorithm is called longest-path algorithm in which the uncovering problem is modeled as a graph. So the essence of longest-path algorithm is to find out the longest path inside the graph. The optimal state sequence which is solution of uncovering problem is constructed from such path.

Keywords: Hidden Markov Model, Uncovering Problem, Longest-path Algorithm

1. Introduction to Hidden Markov Model (HMM)

Markov model (MM) is the statistical model which is used to model the stochastic process. MM is defined as below [1]:

- Given a finite set of state $S = \{s_1, s_2,..., s_n\}$ whose cardinality is n. Let $\prod$ be the *initial state distribution* where $\pi_i \in \prod$ represents the probability that the stochastic process begins in state s_i. We have $\sum_{s_i \in S} \pi_i = 1$.
- The stochastic process is defined as a finite vector $X=(x_1, x_2,..., x_T)$ whose element x_t is a state at time point t. The process X is called *state stochastic process* and $x_t \in S$ equals some state $s_i \in S$. X is also called *state sequence*. The state stochastic process X must meet fully the *Markov property*, namely, given previous state x_{t-1} of process X, the conditional probability of current state x_t is only dependent on the previous state x_{t-1}, not relevant to any further past state $(x_{t-2}, x_{t-3},..., x_1)$. In other words, $P(x_t \mid x_{t-1}, x_{t-2}, x_{t-3},..., x_1) = P(x_t \mid x_{t-1})$ with note that $P(.)$ also denotes probability in this article.
- At each time point, the process changes to the next state based on the *transition probability distribution* a_{ij}, which

depends only on the previous state. So a_{ij} is the probability that the stochastic process changes current state s_i to next state s_j. It means that $a_{ij} = P(x_t=s_j \mid x_{t-1}=s_i) = P(x_{t+1}=s_j \mid x_t=s_i)$. The probability of transitioning from any given state to some next state is 1, we have $\forall s_i \in S, \sum_{s_j \in S} a_{ij} = 1$. All transition probabilities a_{ij} (s) constitute the *transition probability matrix A*. Note that A is n by n matrix because there are n distinct states.

Briefly, MM is the triple $\langle S, A, \prod \rangle$. In typical MM, states are observed directly by users and transition probabilities (A and $\prod$) are unique parameters. Otherwise, hidden Markov model (HMM) is similar to MM except that the underlying states become hidden from observer, they are hidden parameters. HMM adds more output parameters which are called observations. The HMM has further properties as below [1]:

- Suppose there is a finite set of possible observations $\Phi = \{\varphi_1, \varphi_2,..., \varphi_m\}$ whose cardinality is m. There is the second stochastic process which produces *observations* correlating with hidden states. This process is called *observable stochastic process*, which is defined as a finite vector $O = (o_1, o_2,..., o_T)$ whose element o_t is an observation at time point t. Note that $o_t \in \Phi$ equals some φ_k. The process O is often known as *observation se-*

quence.

- There is a probability distribution of producing a given observation in each state. Let $b_i(k)$ be the probability of observation φ_k when the state stochastic process is in state s_i. It means that $b_i(k) = b_i(o_t=\varphi_k) = P(o_t=\varphi_k \mid x_t=s_i)$. The sum of probabilities of all observations which observed in a certain state is 1, we have $\forall s_i \in S, \sum_{\theta_k \in \Phi} b_i(k) = 1$. All probabilities of observations $b_i(k)$ constitute the *observation probability matrix B*. It is convenient for us to use notation b_{ik} instead of notation $b_i(k)$. Note that B is n by m matrix because there are n distinct states and m distinct observations.

Thus, HMM is the 5-tuple $\Delta = \langle S, \Phi, A, B, \prod \rangle$. Note that components S, Φ, A, B, and $\prod$ are often called parameters of HMM in which A, B, and $\prod$ are essential parameters. For example, there are some states of weather: *sunny*, *cloudy*, *rainy* [2, p. 1]. Suppose you need to predict how weather tomorrow is: *sunny*, *cloudy* or *rainy* since you know only observations about the humidity: *dry*, *dryish*, *damp*, *soggy*. We have $S = \{s_1=sunny, s_2=cloudy, s_3=rainy\}$, $\Phi = \{\varphi_1=dry, \varphi_2=dryish, \varphi_3=damp, \varphi_4=soggy\}$. Transition probability matrix A is shown in table 1.

Table 1. Transition probability matrix A.

		Weather current day (Time point *t*)		
		sunny	***cloudy***	***rainy***
Weather previous day (Time point *t* −1)	*sunny*	a_{11}=0.50	a_{12}=0.25	a_{13}=0.25
	cloudy	a_{21}=0.30	a_{22}=0.40	a_{23}=0.30
	rainy	a_{31}=0.25	a_{32}=0.25	a_{33}=0.50

From table 1, we have $a_{11}+a_{12}+a_{13}=1$, $a_{21}+a_{22}+a_{23}=1$, $a_{31}+a_{32}+a_{33}=1$.

Initial state distribution specified as uniform distribution is shown in table 2.

Table 2. Uniform initial state distribution $\prod$.

sunny	cloudy	rainy
π_1=0.33	π_2=0.33	π_3=0.33

From table 2, we have $\pi_1+\pi_2+\pi_3=1$.
Observation probability matrix B is shown in table 3.

Table 3. Observation probability matrix B.

		Humidity			
		dry	***dryish***	***damp***	***soggy***
Weather	*sunny*	b_{11}=0.60	b_{12}=0.20	b_{13}=0.15	b_{14}=0.05
	cloudy	b_{21}=0.25	b_{22}=0.25	b_{23}=0.25	b_{24}=0.25
	rainy	b_{31}=0.05	b_{32}=0.10	b_{33}=0.35	b_{34}=0.50

From table 3, we have $b_{11}+b_{12}+b_{13}+b_{14}=1$, $b_{21}+b_{22}+b_{23}+b_{24}=1$, $b_{31}+b_{32}+b_{33}+b_{34}=1$.

There are three problems of HMM [1] [3, pp. 262-266]:

1. Given HMM Δ and an observation sequence $O = \{o_1, o_2,\ldots, o_T\}$ where $o_t \in \Phi$, how to calculate the probability $P(O|\Delta)$ of this observation sequence. This is *evaluation problem*.

2. Given HMM Δ and an observation sequence $O = \{o_1, o_2,\ldots, o_T\}$ where $o_t \in \Phi$, how to find the state sequence $X = \{x_1, x_2,\ldots, x_T\}$ where $x_t \in S$ so that X is most likely to have produced the observation sequence O. This is *uncovering problem*.

3. Given HMM Δ and an observation sequence $O = \{o_1, o_2,\ldots, o_T\}$ where $o_t \in \Phi$, how to adjust parameters of Δ such as initial state distribution $\prod$, transition probability matrix A, and observation probability matrix B so that the quality of HMM Δ is enhanced. This is *learning problem*.

This article focuses on the uncovering problem. Section 2 mentions some methods to solve the uncovering problem, in which Viterbi is the best method. Section 3 is the main one that proposes the longest-path algorithm.

2. HMM Uncovering Problem

According to uncovering problem, it is required to establish an *optimal criterion* so that the state sequence $X = \{x_1, x_2,\ldots, x_T\}$ leads to maximizing such criterion. The simple criterion is the conditional probability of sequence X with respect to sequence O and model Δ, denoted $P(X|O,\Delta)$. We can apply brute-force strategy: "go through all possible such X and pick the one leading to maximizing the criterion $P(X|O,\Delta)$".

$$X = \underset{X}{\operatorname{argmax}}\big(P(X|O, \Delta)\big)$$

This strategy is impossible if the number of states and observations is huge. Another popular way is to establish a so-called *individually optimal criterion* [3, p. 263] which is described right later.

Let $\gamma_t(i)$ be joint probability that the stochastic process is in state s_i at time point t with observation sequence $O = \{o_1, o_2,\ldots, o_T\}$, equation (1) specifies this probability based on forward variable α_t and backward variable β_t. Please read [3, pp. 262-263] to comprehend α_t and β_t. The variable $\gamma_t(i)$ is also called *individually optimal criterion*.

$$\gamma_t(i) = P(o_1, o_2, \ldots, o_T, x_t = s_i|\Delta) = \alpha_t(i)\beta_t(i) \qquad (1)$$

Because the probability $P(o_1, o_2, \ldots, o_T|\Delta)$ is not relevant to state sequence X, it is possible to remove it from the optimization criterion. Thus, equation (2) specifies how to find out the optimal state x_t of X at time point t.

$$x_t = \underset{i}{\operatorname{argmax}} \gamma_t(i) = \underset{i}{\operatorname{argmax}} \alpha_t(i)\beta_t(i) \qquad (2)$$

The procedure to find out state sequence $X = \{x_1, x_2,\ldots, x_T\}$ based on individually optimal criterion is called *individually optimal procedure* that includes three steps, shown in table 4.

Table 4. Viterbi algorithm to solve uncovering problem.

1. Initialization step:
   - Initializing $\alpha_1(i) = b_i(o_1)\pi_i$ for all $1 \leq i \leq n$
   - Initializing $\beta_T(i) = 1$ for all $1 \leq i \leq n$
2. Recurrence step:
   - Calculating all $\alpha_{t+1}(i)$ for all $1 \leq i \leq n$ and $1 \leq t \leq T-1$
   - Calculating all $\beta_t(i)$ for all $1 \leq i \leq n$ and $t=T-1, t=T-2,\ldots, t=1$
   - Calculating all $\gamma_t(i)=\alpha_t(i)\beta_t(i)$ for all $1 \leq i \leq n$ and $1 \leq t \leq T$
   - Determining optimal state x_t of X at time point t is the one that maximizes $\gamma_t(i)$ over all values s_i.
     $$x_t = \underset{i}{\operatorname{argmax}} \gamma_t(i)$$
3. Final step: The state sequence $X = \{x_1, x_2,\ldots, x_T\}$ is totally determined when its partial states x_t (s) where $1 \leq t \leq T$ are found in recurrence step.

It is required to execute $n+(5n^2-n)(T-1)+2nT$ operations for individually optimal procedure due to:

- There are n multiplications for calculating $\alpha_1(i)$ (s).
- The recurrence step runs over $T-1$ times. There are $2n^2(T-1)$ operations for determining $\alpha_{t+1}(i)$ (s) over all $1 \le i \le n$ and $1 \le t \le T-1$. There are $(3n-1)n(T-1)$ operations for determining $\beta_t(i)$ (s) over all $1 \le i \le n$ and $t=T-1, t=T-2,\dots, t=1$. There are nT multiplications for determining $\gamma_t(i)=\alpha_t(i)\beta_t(i)$ over all $1 \le i \le n$ and $1 \le t \le T$. There are nT comparisons for determining optimal state $x_t = \operatorname*{argmax}_i \gamma_t(i)$ over all $1 \le i \le n$ and $1 \le t \le T$. In general, there are $2n^2(T-1)+ (3n-1)n(T-1) + nT + nT = (5n^2-n)(T-1) + 2nT$ operations at the recurrence step.

Inside $n + (5n^2-n)(T-1) + 2nT$ operations, there are $n + (n+1)n(T-1) + 2n^2(T-1) + nT = (3n^2+n)(T-1) + nT + n$ multiplications and $(n-1)n(T-1) + (n-1)n(T-1) = 2(n^2-n)(T-1)$ additions and nT comparisons.

The individually optimal criterion $\gamma_t(i)$ does not reflect the whole probability of state sequence X given observation sequence O because it focuses only on how to find out each partially optimal state x_t at each time point t. Thus, the individually optimal procedure is heuristic method. Viterbi algorithm [3, p. 264] is alternative method that takes interest in the whole state sequence X by using joint probability $P(X,O|\Delta)$ of state sequence and observation sequence as optimal criterion for determining state sequence X. Let $\delta_t(i)$ be the maximum joint probability of observation sequence O and state $x_t=s_i$ over $t-1$ previous states. The quantity $\delta_t(i)$ is called *joint optimal criterion* at time point t, which is specified by (3).

$$\delta_t(i) = \max_{x_1,x_2,\dots,x_{t-1}}\left(P(o_1, o_2, \dots, o_t, x_1, x_2, \dots, x_t = s_i|\Delta)\right) \quad (3)$$

The recurrence property of *joint optimal criterion* is specified by (4).

$$\delta_{t+1}(j) = \left(\max_i\left(\delta_t(i)a_{ij}\right)\right)b_j(o_{t+1}) \quad (4)$$

Given criterion $\delta_{t+1}(j)$, the state $x_{t+1}=s_j$ that maximizes $\delta_{t+1}(j)$ is stored in the backtracking state $q_{t+1}(j)$ that is specified by (5).

$$q_{t+1}(j) = \operatorname*{argmax}_i\left(\delta_t(i)a_{ij}\right) \quad (5)$$

Note that index i is identified with state $s_i \in S$ according to (5). The Viterbi algorithm based on joint optimal criterion $\delta_t(i)$ includes three steps described in table 5.

Table 5. *Viterbi algorithm to solve uncovering problem.*

1. Initialization step:
   - Initializing $\delta_1(i) = b_i(o_1)\pi_i$ for all $1 \le i \le n$
   - Initializing $q_1(i) = 0$ for all $1 \le i \le n$
2. Recurrence step:
   - Calculating all $\delta_{t+1}(j) = \left(\max_i\left(\delta_t(i)a_{ij}\right)\right)b_j(o_{t+1})$ for all $1 \le i,j \le n$ and $1 \le t \le T-1$ according to (4).
   - Keeping tracking optimal states $q_{t+1}(j) = \operatorname*{argmax}_i\left(\delta_t(i)a_{ij}\right)$ for all $1 \le j \le n$ and $1 \le t \le T-1$ according to (5).
3. State sequence backtracking step: The resulted state sequence $X = \{x_1, x_2,\dots, x_T\}$ is determined as follows:
   - The last state $x_T = \operatorname*{argmax}_j(\delta_T(j))$
   - Previous states are determined by backtracking: $x_t = q_{t+1}(x_{t+1})$ for $t=T-1, t=T-2,\dots, t=1$.

The total number of operations inside the Viterbi algorithm is $2n+(2n^2+n)(T-1)$ as follows:

- There are n multiplications for initializing n values $\delta_1(i)$ when each $\delta_1(i)$ requires 1 multiplication.
- There are $(2n^2+n)(T-1)$ operations over the recurrence step because there are $n(T-1)$ values $\delta_{t+1}(j)$ and each $\delta_{t+1}(j)$ requires n multiplications and n comparisons for maximizing $\max_i\left(\delta_t(i)a_{ij}\right)$ plus 1 multiplication.
- There are n comparisons for constructing the state sequence X, $x_T = \max_j\left(q_T(j)\right)$.

Inside $2n+(2n^2+n)(T-1)$ operations, there are $n+(n^2+n)(T-1)$ multiplications and $n^2(T-1)+n$ comparisons. The number of operations with regard to Viterbi algorithm is smaller than the number of operations with regard to individually optimal procedure when individually optimal procedure requires $(5n^2-n)(T-1)+2nT+n$ operations. Therefore, Viterbi algorithm is more effective than individually optimal procedure. Besides, individually optimal procedure does not reflect the whole probability of state sequence X given observation sequence O. The successive section describes longest-path algorithm which is a competitor of Viterbi.

3. Longest-path Algorithm to Solve HMM Uncovering Problem

Essentially, Viterbi algorithm maximizes the joint probability $P(X, O|\Delta)$ instead of maximizing the conditional probability $P(X|O, \Delta)$. I propose so-called *longest-path algorithm* based on longest path of graph for solving uncovering problem. This algorithm that maintains using the conditional probability $P(X|O, \Delta)$ as optimal criterion gives a viewpoint different from the viewpoint of Viterbi algorithm although it is easy for you to recognize that the ideology of the longest-path algorithm does not go beyond the ideology of Viterbi algorithm after you comprehend the longest-path algorithm. Following is description of longest-path algorithm.

The optimal criterion $P(X|O, \Delta)$ of graphic method is:

$$P(X|O, \Delta) = P(x_1, x_2, \dots, x_T|o_1, o_2, \dots, o_T)$$
$$= P(x_1, x_2, \dots, x_{T-1}, x_T|o_1, o_2, \dots, o_{T-1}, o_T)$$

$$= P(x_T|x_1, x_2, \dots, x_{T-1}, o_1, o_2, \dots, o_{T-1}, o_T)$$
$$* P(x_1, x_2, \dots, x_{T-1}|o_1, o_2, \dots, o_{T-1}, o_T)$$

(Due to multiplication rule)

$$= P(x_T|x_{T-1}, o_1, o_2, \dots, o_{T-1}, o_T)$$
$$* P(x_1, x_2, \dots, x_{T-1}|o_1, o_2, \dots, o_{T-1}, o_T)$$

(Due to Markov property: the probability of current state is only dependent on the probability of right previous state)

$$= P(x_T|x_{T-1}, o_T) * P(x_1, x_2, \dots, x_{T-1}|o_1, o_2, \dots, o_{T-1})$$

(Because an observation is only dependent on the time point when it is observed)

By recurrence calculation on probability

$$P(x_1, x_2, \dots, x_{T-1}|o_1, o_2, \dots, o_{T-1})$$

We have:

$$P(X|O, \Delta) = P(x_1, x_2, \ldots, x_T | o_1, o_2, \ldots, o_T)$$
$$= P(x_1|o_1)P(x_2|x_1, o_2) \ldots P(x_t|x_{t-1}, o_t) \ldots P(x_T|x_{T-1}, o_T)$$

Applying Bayes' rule into the probability $P(x_{t-1}, x_t|o_t)$, we have:

$$P(x_{t-1}, x_t|o_t) = \frac{P(x_t|x_{t-1}, o_t)P(x_{t-1}|o_t)}{P(x_t|o_t)}$$

$$= P(x_t|x_{t-1}, o_t)\frac{1}{P(x_t|o_t)}P(x_{t-1}|o_t)$$

$$= P(x_t|x_{t-1}, o_t)\frac{1}{P(x_t|o_t)}\frac{P(o_t|x_{t-1})P(x_{t-1})}{P(o_t)}$$

(Applying Bayes' rule into the probability $P(x_{t-1}|o_t)$)

$$= P(x_t|x_{t-1}, o_t)\frac{P(o_t)}{P(o_t|x_t)P(x_t)}\frac{P(o_t|x_{t-1})P(x_{t-1})}{P(o_t)}$$

(Applying Bayes' rule into the probability $P(x_t|o_t)$)

$$= \frac{P(x_t|x_{t-1}, o_t)P(o_t|x_{t-1})P(x_{t-1})}{P(o_t|x_t)P(x_t)}$$
$$= \frac{P(x_t|x_{t-1}, o_t)P(o_t)P(x_{t-1})}{P(o_t|x_t)P(x_t)}$$

(Because an observation is only dependent on the time point when it is observed, $P(o_t|x_{t-1}) = P(o_t)$)

Applying Bayes' rule into the probability $P(x_{t-1}, x_t|o_t)$ by another way, we have:

$$P(x_{t-1}, x_t|o_t) = \frac{P(o_t|x_{t-1}, x_t)P(x_{t-1}, x_t)}{P(o_t)}$$

$$= \frac{P(o_t|x_t)P(x_{t-1}, x_t)}{P(o_t)}$$

(Because an observation is only dependent on the time point when it is observed, $P(o_t|x_{t-1}, x_t) = P(o_t|x_t)$)

$$= \frac{P(o_t|x_t)P(x_t|x_{t-1})P(x_{t-1})}{P(o_t)}$$

(Applying multiplication rule into the probability $P(x_t|o_t)$)
Because we had

$$P(x_{t-1}, x_t|o_t) = \frac{P(x_t|x_{t-1}, o_t)P(o_t)P(x_{t-1})}{P(o_t|x_t)P(x_t)}$$

It implies that

$$\frac{P(x_t|x_{t-1}, o_t)P(o_t)P(x_{t-1})}{P(o_t|x_t)P(x_t)} = \frac{P(o_t|x_t)P(x_t|x_{t-1})P(x_{t-1})}{P(o_t)}$$

$$\Rightarrow P(x_t|x_{t-1}, o_t) = \frac{\left(P(o_t|x_t)\right)^2 P(x_t|x_{t-1})P(x_t)}{\left(P(o_t)\right)^2}$$

We have

$$P(X|O, \Delta) = P(x_1, x_2, \ldots, x_T | o_1, o_2, \ldots, o_T)$$
$$= P(x_1|o_1)P(x_2|x_1, o_2) \ldots P(x_{t-1}, x_t|o_t) \ldots P(x_T|x_{T-1}, o_T)$$

$$= \frac{P(o_1|x_1)P(x_1)}{P(o_1)} * \frac{\left(P(o_2|x_2)\right)^2 P(x_2|x_1)P(x_2)}{\left(P(o_2)\right)^2} * \cdots$$

$$* \frac{\left(P(o_t|x_t)\right)^2 P(x_t|x_{t-1})P(x_t)}{\left(P(o_t)\right)^2} * \cdots$$

$$* \frac{\left(P(o_T|x_T)\right)^2 P(x_T|x_{T-1})P(x_T)}{\left(P(o_t)\right)^2}$$

$$= \frac{P(o_1)}{\left(P(o_1)\right)^2\left(P(o_2)\right)^2 \cdots \left(P(o_t)\right)^2 \cdots \left(P(o_T)\right)^2}$$

$$* P(o_1|x_1)P(x_1) * \left(P(o_2|x_2)\right)^2 P(x_2|x_1)P(x_2) * \cdots$$

$$* \left(P(o_t|x_t)\right)^2 P(x_t|x_{t-1})P(x_t) * \cdots$$

$$* \left(P(o_T|x_T)\right)^2 P(x_T|x_{T-1})P(x_T)$$

$$= c w_1 w_2 \ldots w_t \ldots w_T$$

Where,

$$\begin{cases} c \text{ is constant} \\ c = \dfrac{P(o_1)}{\left(P(o_1)\right)^2\left(P(o_2)\right)^2 \cdots \left(P(o_t)\right)^2 \cdots \left(P(o_T)\right)^2} \\ w_1 = P(o_1|x_1)P(x_1) \text{ when } t = 1 \\ w_t = \left(P(o_t|x_t)\right)^2 P(x_t|x_{t-1})P(x_t), \forall 1 < t \leq T \end{cases}$$

Because the constant c is independent from state transitions, maximizing the criterion $P(X|O, \Delta)$ with regard to state transitions is the same to maximizing the product $w_1 w_2 \ldots w_t \ldots w_T$. Let ρ be this product and so, ρ is the optimal criterion of longest-path algorithm, re-written by (6).

$$\rho = w_1 w_2 \ldots w_t \ldots w_T \qquad (6)$$

Where,

$$\begin{cases} w_1 = P(o_1|x_1)P(x_1) \text{ when } t = 1 \\ w_t = \left(P(o_t|x_t)\right)^2 P(x_t|x_{t-1})P(x_t), \forall 1 < t \leq T \end{cases}$$

The essence of longest-path algorithm is to construct a graph and then, the algorithm finds out the longest path inside such graph with attention that the optimal criterion ρ represents length of every path inside the graph. There is an interesting thing that such length ρ is product of weights instead of sequence of additions as usual. The criterion ρ is function of state transitions and the longest-path algorithm aims to maximize ρ. Following is description of how to build up the graph.

Each w_t representing the influence of state x_t on the observation sequence $O = \{o_1, o_2, \ldots, o_T\}$ at time point t is dependent on states x_{t-1} and x_t. We will create a graph from these $w_t (s)$. Because there are n possible values of x_t, the state x_t is de-

composed into n nodes $X_{t1}, X_{t2}, \ldots, X_{tn}$. There are T time points, we have nT time nodes. Let $X = \{X_0, X_{11}, X_{12}, \ldots, X_{1n}, X_{21}, X_{22}, \ldots, X_{2n}, \ldots, X_{T1}, X_{T2}, \ldots, X_{Tn}\}$ be a set of $1+nT$ nodes where X_0 is null node. Firstly, we create n weighted arcs from node X_0 to n nodes $X_{11}, X_{12}, \ldots, X_{1n}$ at the first time point. These directed arcs are denoted $W_{0111}, W_{0112}, \ldots, W_{011n}$ and their weights are also denoted $W_{0111}, W_{0112}, \ldots, W_{011n}$. These weights W_{011j} (s) at the first time point are calculated according to w_1 (see (6)). Equation (7) determines W_{1j} (s).

$$W_{011j} = P(o_1|x_1 = s_j)P(x_1 = s_j) = b_j(o_1)\pi_j$$

$$\forall j = \overline{1, n} \qquad (7)$$

Your attention please, it is conventional that W_{0i1j} is equal to W_{011j}, $\forall i = \overline{1, n}$ because the null node X_0 has no state.

$$W_{0i1j} = W_{011j}, \forall i = \overline{1, n}$$

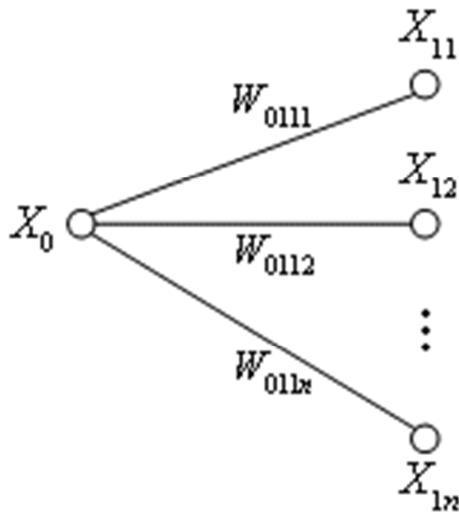Moreover, these weights W_{011j} (s) are depicted by fig. 1.

Figure 1. *Weighted arcs from null node X_0 to n nodes $X_{11}, X_{12}, \ldots, X_{1n}$.*

For example, given weather HMM Δ whose parameters A, B, and $\prod$ are specified in tables 1, 2, and 3, suppose observation sequence is $O = \{o_1=\varphi_4=soggy, o_2=\varphi_1=dry, o_3=\varphi_2=dryish\}$, we have 3 weights at the initial time point as follows:

$$W_{0111} = b_1(o_1 = \varphi_4)\pi_1 = b_{14}\pi_1$$

$$W_{0112} = b_2(o_1 = \varphi_4)\pi_2 = b_{24}\pi_2$$

$$W_{0113} = b_3(o_1 = \varphi_4)\pi_3 = b_{34}\pi_3$$

For each node $X_{(t-1)i}$ where $t > 1$, we create n weighted arcs from node $X_{(t-1)i}$ to n nodes $X_{t1}, X_{t2}, \ldots, X_{tn}$ at the time point t. These directed arcs are denoted $W_{(t-1)it1}, W_{(t-1)it2}, \ldots, W_{(t-1)itn}$ and their weights are also denoted $W_{(t-1)it1}, W_{(t-1)it2}, \ldots, W_{(t-1)itn}$. These weights $W_{(t-1)itj}$ at the time point t are calculated according to w_t (see (6)). Equation (8) determines $W_{(t-1)itj}$.

$$W_{(t-1)itj} = \left(P(o_t|x_t = s_j)\right)^2 * P(x_t = s_j|x_{t-1} = s_i)$$

$$* P(x_t = s_j) = \left(b_j(o_t)\right)^2 a_{ij}\pi_j$$

$$\forall i, j = \overline{1, n} \qquad (8)$$

Moreover, these weights $W_{(t-1)itj}$ (s) are depicted by fig. 2.
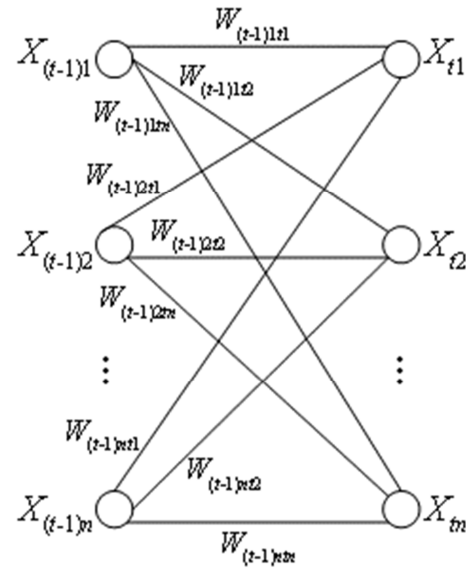

Figure 2. *Weighted arcs from n node $X_{(t-1)i}$ to n nodes X_{tj} at time point t.*

Going back given weather HMM Δ whose parameters A, B, and $\prod$ are specified in tables 1, 2, and 3, suppose observation sequence is $O = \{o_1=\varphi_4=soggy, o_2=\varphi_1=dry, o_3=\varphi_2=dryish\}$, we have 18 weights from time point 1 to time point 3 as follows:

$$W_{1121} = \left(b_1(o_2 = \varphi_1)\right)^2 a_{11}\pi_1 = (b_{11})^2 a_{11}\pi_1$$

$$W_{1122} = \left(b_2(o_2 = \varphi_1)\right)^2 a_{12}\pi_2 = (b_{21})^2 a_{12}\pi_2$$

$$W_{1123} = \left(b_3(o_2 = \varphi_1)\right)^2 a_{13}\pi_3 = (b_{31})^2 a_{13}\pi_3$$

$$W_{1221} = \left(b_1(o_2 = \varphi_1)\right)^2 a_{21}\pi_1 = (b_{11})^2 a_{21}\pi_1$$

$$W_{1222} = \left(b_2(o_2 = \varphi_1)\right)^2 a_{22}\pi_2 = (b_{21})^2 a_{22}\pi_2$$

$$W_{1223} = \left(b_3(o_2 = \varphi_1)\right)^2 a_{23}\pi_3 = (b_{31})^2 a_{23}\pi_3$$

$$W_{1321} = \left(b_1(o_2 = \varphi_1)\right)^2 a_{31}\pi_1 = (b_{11})^2 a_{31}\pi_1$$

$$W_{1322} = \left(b_2(o_2 = \varphi_1)\right)^2 a_{32}\pi_2 = (b_{21})^2 a_{32}\pi_2$$

$$W_{1323} = \left(b_3(o_2 = \varphi_1)\right)^2 a_{33}\pi_3 = (b_{31})^2 a_{33}\pi_3$$

$$W_{2131} = \left(b_1(o_3 = \varphi_2)\right)^2 a_{11}\pi_1 = (b_{12})^2 a_{11}\pi_1$$

$$W_{2132} = \left(b_2(o_3 = \varphi_2)\right)^2 a_{12}\pi_2 = (b_{22})^2 a_{12}\pi_2$$

$$W_{2133} = \left(b_3(o_3 = \varphi_2)\right)^2 a_{13}\pi_3 = (b_{32})^2 a_{13}\pi_3$$

$$W_{2231} = \left(b_1(o_3 = \varphi_2)\right)^2 a_{21}\pi_1 = (b_{12})^2 a_{21}\pi_1$$

$$W_{2232} = \left(b_2(o_3 = \varphi_2)\right)^2 a_{22}\pi_2 = (b_{22})^2 a_{22}\pi_2$$

$$W_{2233} = \left(b_3(o_3 = \varphi_2)\right)^2 a_{23}\pi_3 = (b_{32})^2 a_{23}\pi_3$$

$$W_{2331} = \left(b_1(o_3 = \varphi_2)\right)^2 a_{31}\pi_1 = (b_{12})^2 a_{31}\pi_1$$

$$W_{2332} = \left(b_2(o_3 = \varphi_2)\right)^2 a_{32}\pi_2 = (b_{22})^2 a_{32}\pi_2$$

$$W_{2333} = \left(b_3(o_3 = \varphi_2)\right)^2 a_{33}\pi_3 = (b_{32})^2 a_{33}\pi_3$$

In general, there are $(T–1)n^2$ weights from time point 1 to time point T. Moreover, there are n weights derived from null node X_0 at time point 1. Let W be set of $n+(T–1)n^2$ weights from null node X_0 to nodes $X_{T1}, X_{T2},…, X_{Tn}$ at the last time point T. Let $G = <X, W>$ be the graph consisting of the set of nodes $X = \{X_0, X_{11}, X_{12},…, X_{1n}, X_{21}, X_{22},…, X_{2n},…, X_{T1}, X_{T2},…, X_{Tn}\}$ be a set of $n+(T–1)n^2$ weights W. The graph G is called *state transition graph* shown in fig. 3.
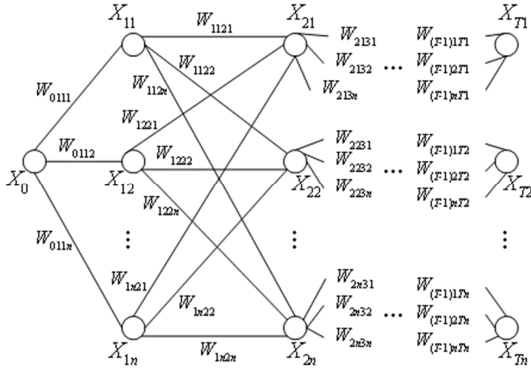

Figure 3. State transition graph.

Please pay attention to a very important thing that both graph G and its weights are not determined before the longest-path algorithm is executed because there are a huge number of nodes and arcs. State transition graph shown in fig. 3 is only illustrative example. Going back given weather HMM Δ whose parameters A, B, and $\prod$ are specified in tables 1, 2, and 3, suppose observation sequence is $O = \{o_1=\varphi_4=soggy, o_2=\varphi_1=dry, o_3=\varphi_2=dryish\}$, the state transition graph of this weather example is shown in fig. 4.
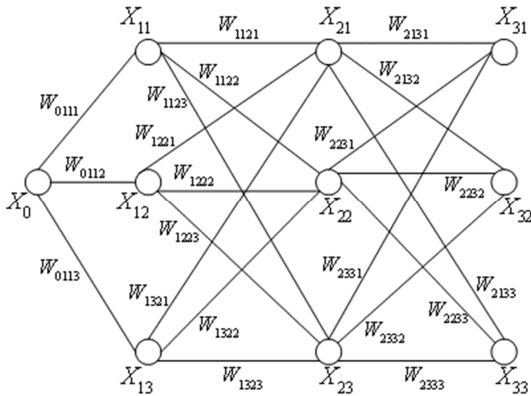

Figure 4. State transition graph of weather example.

The ideology of the longest-path algorithm is to solve uncovering problem by finding the longest path of state transition graph where the whole length of every path is represented by the optimal criterion ρ (see (6)). In other words, the longest-path algorithm maximizes the optimal criterion ρ by finding the longest path. Let $X = \{x_1, x_2,…, x_T\}$ be the longest path of state transition graph and so, length of X is maximum value of the path length ρ. The path length ρ is calculated as

product of weights $W_{(t–1)itj}$ (s). By heuristic assumption, ρ is maximized locally by maximizing weights $W_{(t–1)itj}$ (s) at each time point. The longest-path algorithm is described by pseudo-code shown in table 6 with note that X is state sequence that is ultimate result of the longest-path algorithm.

Table 6. Longest-path algorithm.

X is initialized to be empty, $X = \emptyset$.
Calculating initial weights $W_{0111}, W_{0112},…, W_{011n}$ according to (7). $\qquad j = \underset{k}{\text{argmax}}\{W_{011k}\}$ where $k = \overline{1,n}$ Adding state $x_1=s_j$ to the longest path: $X = X \cup \{x_1 = s_j\}$
For $t = 2$ to T $\quad$ Calculating n weights $W_{(t–1)jt1}, W_{(t–1)jt2},…, W_{(t–1)jtn}$ according to (8). $\qquad j = \underset{k}{\text{argmax}}\{W_{(t–1)jtk}\}$ where $k = \overline{1,n}$ $\quad$ Adding state $x_t=s_j$ to the longest path: $X = X \cup \{x_t = s_j\}$ End for

The total number of operations inside the longest-path algorithm is $2n+4n(T–1)$ as follows:
- There are n multiplications for initializing n weights $W_{0111}, W_{0112},…, W_{011n}$ when each weight W_{011j} requires 1 multiplication. There are n comparisons due to finding maximum weight index $j = \text{argmax}_k\{W_{011k}\}$.
- There are $3n(T–1)$ multiplications over the loop inside the algorithm because there are $n(T–1)$ weights $W_{(t–1)jtk}$ over the loop and each $W_{(t–1)jtk}$ requires 3 multiplications. There are $n(T–1)$ comparisons over the loop inside the algorithm due to finding maximum weight indices: $j = \text{argmax}_k\{W_{(t–1)jtk}\}$.

Inside $2n+4n(T–1)$ operations, there are $n+3n(T–1)$ multiplications and $n+n(T–1)$ comparisons.

The longest-path algorithm is similar to Viterbi algorithm (see table 5) with regard to the aspect that the path length ρ is calculated accumulatively but computational formulas and viewpoints of longest-path algorithm and Viterbi algorithm are different. The longest-path algorithm is more effective than Viterbi algorithm because it requires $2n+4n(T–1)$ operations while Viterbi algorithm executes $2n+(2n^2+n)(T–1)$ operations. However, longest-path algorithm does not produce the most accurate result because the path length ρ is maximized locally by maximizing weights $W_{(t–1)itj}$ (s) at each time point, which leads that the resulted sequence X may not be global longest path. In general, the longest-path algorithm is heuristic algorithm that gives a new viewpoint of uncovering problem when applying graphic approach into solving uncovering problem.

Going back given weather HMM Δ whose parameters A, B, and $\prod$ are specified in tables 1, 2, and 3, suppose observation sequence is $O = \{o_1=\varphi_4=soggy, o_2=\varphi_1=dry, o_3=\varphi_2=dryish\}$, the longest-path algorithm is applied to find out the optimal state sequence $X = \{x_1, x_2, x_3\}$ as below.

At the first time point, we have:

$$W_{0111} = b_{14}\pi_1 = 0.05 * 0.33 = 0.0165$$

$$W_{0112} = b_{24}\pi_2 = 0.25 * 0.33 = 0.0825$$

$$W_{0113} = b_{34}\pi_3 = 0.5 * 0.33 = 0.165$$

$$j = \underset{k}{\arg\max}\{W_{011k}\} = \underset{k}{\arg\max}\{W_{0111}, W_{0112}, W_{0113}\} = 3$$

$$X = X \cup \{x_1 = s_j\} = X \cup \{x_1 = s_3\} = \{x_1 = rainy\}$$

At the second time point, we have:

$$W_{1j21} = W_{1321} = (b_{11})^2 a_{31} \pi_1 = 0.6^2 * 0.25 * 0.33$$
$$= 0.0297$$

$$W_{1j22} = W_{1322} = (b_{21})^2 a_{32} \pi_2 = 0.25^2 * 0.25 * 0.33$$
$$= 0.00515625$$

$$W_{1j23} = W_{1323} = (b_{31})^2 a_{33} \pi_3 = 0.05^2 * 0.5 * 0.33$$
$$= 0.0004125$$

$$j = \underset{k}{\arg\max}\{W_{132k}\} = \underset{k}{\arg\max}\{W_{1321}, W_{1322}, W_{1323}\} = 1$$

$$X = X \cup \{x_2 = s_j\} = X \cup \{x_2 = s_1\}$$
$$= \{x_1 = rainy, x_2 = sunny\}$$

At the third time point, we have:

$$W_{2j31} = W_{2131} = (b_{12})^2 a_{11} \pi_1 = 0.2^2 * 0.5 * 0.33$$
$$= 0.0066$$

$$W_{2j32} = W_{2132} = (b_{22})^2 a_{12} \pi_2 = 0.25^2 * 0.25 * 0.33$$
$$= 0.00515625$$

$$W_{2j33} = W_{2133} = (b_{32})^2 a_{13} \pi_3 = 0.1^2 * 0.25 * 0.33$$
$$= 0.000825$$

$$j = \underset{k}{\arg\max}\{W_{213k}\} = \underset{k}{\arg\max}\{W_{2131}, W_{2132}, W_{2133}\} = 1$$

$$X = X \cup \{x_3 = s_j\} = X \cup \{x_3 = s_1\}$$
$$= \{x_1 = rainy, x_2 = sunny, x_3 = sunny\}$$

As a result, the optimal state sequence is $X = \{x_1=rainy, x_2=sunny, x_3=sunny\}$. The result from the longest-path algorithm in this example is the same to the one from individually optimal procedure (see table 4) and Viterbi algorithm (see table 5).

The longest-path algorithm does not result out accurate state sequence X because it assumes that two successive nodes $X_{(t-1)i}$ and X_{tj} are mutually independent, which leads that the path length ρ is maximized locally by maximizing weight $W_{(t-1)itj}$ at each time point, while equation (6) indicates that the former node $X_{(t-1)i}$ is dependent on the prior node X_{tj}. However, according to Markov property, two intermittent nodes $X_{(t-1)i}$ and $X_{(t+1)k}$ are conditional independent given the middle node X_{tj}. This observation is very important, which help us to enhance the accuracy of longest-path algorithm. The advanced longest-path algorithm divides the path represented by ρ into a set of 2-weight intervals. Each 2-weight interval includes two successive weights $W_{(t-1)itj}$ and $W_{ti(t+1)j}$ corresponding three nodes $X_{(t-1)i}$, X_{tj}, and $X_{(t+1)k}$ where the middle node X_{tj} is also called the *midpoint* of 2-weight interval. The advanced longest-path algorithm maximizes the path ρ by maximizing every

2-weight interval. Each 2-weight interval has $2n^2$ connections (sub-paths) because each weight $W_{(t-1)itj}$ or $W_{ti(t+1)j}$ has n^2 values. Fig. 5 depicts an example of 2-weight interval.
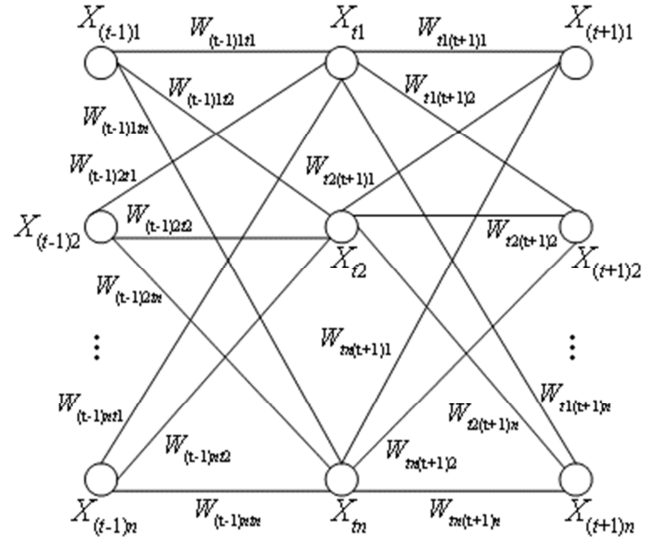

Figure 5. The 2-weight interval.

The advanced longest-path algorithm is described by pseudo-code shown in table 7.

Table 7. Advanced longest-path algorithm.

X is initialized to be empty, $X = \emptyset$.

$i = 1$
For $t = 1$ to T step 2
    // Note that time point t is increased by 2 as follows: 1, 3, 5,...
    Calculating n weights $W_{(t-1)it1}, W_{(t-1)it2},\ldots, W_{(t-1)itn}$ according to (7) and (8).

    For $j = 1$ to n
        Calculating n weights $W_{tj(t+1)1}, W_{tj(t+1)2},\ldots, W_{tj(t+1)n}$ according to (8).
$$k_j = \underset{l}{\arg\max}\{W_{tj(t+1)l}\}$$
    End for
$$u = \underset{j}{\arg\max}\left(W_{(t-1)itj} W_{tj(t+1)k_j}\right)$$
    Adding two states $x_t = s_u$ and $x_{t+1} = s_{k_u}$ to the longest path: $X = X \cup \{x_t = s_u\} \cup \{x_{t+1} = s_v\}$

    $i = k_u$
End for

Because two intermittent nodes $X_{(t-1)i}$ and $X_{(t+1)k}$ that are two end-points of a 2-weight interval are conditional independent given the midpoint X_{tj}, the essence of advanced longest-path algorithm is to adjust the midpoint of 2-weight interval so as to maximize such 2-weight interval.

The total number of operations inside the longest-path algorithm is $(2n^2+1.5n)T$ as follows:

- There are n multiplications for determining weights $W_{(t-1)it1}, W_{(t-1)it2},\ldots, W_{(t-1)itn}$. Shortly, there are $nT/2 = 0.5nT$ multiplications over the whole algorithm because time point is increased by 2.
- There are $3n^2$ multiplications for determining n^2 weights $W_{tj(t+1)l}$ (s) at each time point when each weight requires 3 multiplications. There are n multiplications for determining product $W_{(t-1)itj}W_{tj(t+1)k_j}$. Shortly, there are

$(3n^2+n)T/2 = (1.5n^2+0.5n)T$ multiplications over the whole algorithm because time point is increased by 2.

- There are n^2+n comparisons for maximizing: $\text{argmax}_l\{W_{tj(t+1)l}\}$ and $\text{argmax}_{j,k}\left(W_{(t-1)itj}W_k\right)$. Shortly, there are $(n^2+n)T/2 = (0.5n^2+0.5n)T$ multiplications over the whole algorithm because time point is increased by 2.

Inside $(2n^2+1.5n)T$ operations, there are $(1.5n^2+n)T$ multiplications and $(0.5n^2+0.5n)T$ comparisons. The advanced longest-path algorithm is not more effective than Viterbi algorithm because it requires $(2n^2+1.5n)T$ operations while Viterbi algorithm executes $2n+(2n^2+n)(T-1)$ operations but it is more accurate than normal longest-path algorithm aforementioned in table 6.

Going back given weather HMM Δ whose parameters A, B, and $\prod$ are specified in tables 1, 2, and 3, suppose observation sequence is $O = \{o_1=\varphi_4=soggy, \ o_2=\varphi_1=dry, \ o_3=\varphi_2=dryish\}$, the advanced longest-path algorithm is applied to find out the optimal state sequence $X = \{x_1, x_2, x_3\}$ as follows:

At $t=1$, we have:

$$W_{0111} = b_{14}\pi_1 = 0.05 * 0.33 = 0.0165$$

$$W_{0112} = b_{24}\pi_2 = 0.25 * 0.33 = 0.0825$$

$$W_{0113} = b_{34}\pi_3 = 0.5 * 0.33 = 0.165$$

$$W_{1121} = (b_{11})^2 a_{11}\pi_1 = 0.6^2 * 0.5 * 0.33 = 0.0594$$

$$W_{1122} = (b_{21})^2 a_{12}\pi_2 = 0.25^2 * 0.25 * 0.33 = 0.00515625$$

$$W_{1123} = (b_{31})^2 a_{13}\pi_3 = 0.05^2 * 0.25 * 0.33 = 0.00020625$$

$$k_1 = \text{argmax}_l\{W_{112l}\} = \text{argmax}_l\{W_{1121}, W_{1122}, W_{1123}\} = 1$$

$$W_{0111}W_{112k_1} = W_{0111}W_{1121} = 0.0165 * 0.0594 = 0.0009801$$

$$W_{1221} = (b_{11})^2 a_{21}\pi_1 = 0.6^2 * 0.3 * 0.33 = 0.03564$$

$$W_{1222} = (b_{21})^2 a_{22}\pi_2 = 0.25^2 * 0.4 * 0.33 = 0.00825$$

$$W_{1223} = (b_{31})^2 a_{23}\pi_3 = 0.05^2 * 0.3 * 0.33 = 0.0002475$$

$$k_2 = \text{argmax}_l\{W_{122l}\} = \text{argmax}_l\{W_{1221}, W_{1222}, W_{1223}\} = 1$$

$$W_{0112}W_{122k_2} = W_{0112}W_{1221} = 0.0825 * 0.03564 = 0.0029403$$

$$W_{1321} = (b_{11})^2 a_{31}\pi_1 = 0.6^2 * 0.25 * 0.33 = 0.0297$$

$$W_{1322} = (b_{21})^2 a_{32}\pi_2 = 0.25^2 * 0.25 * 0.33 = 0.00515625$$

$$W_{1323} = (b_{31})^2 a_{33}\pi_3 = 0.05^2 * 0.5 * 0.33 = 0.0004125$$

$$k_3 = \text{argmax}_l\{W_{132l}\} = \text{argmax}_l\{W_{1321}, W_{1322}, W_{1323}\} = 1$$

$$W_{0113}W_{132k_3} = W_{0113}W_{1321} = 0.165 * 0.0297 = 0.0049005$$

$$u = \text{argmax}_j\left\{W_{011j}W_{112k_j}\right\}$$
$$= \text{argmax}_j\{W_{0111}W_{112k_1}, W_{0112}W_{122k_2}, W_{0113}W_{132k_3}\} = 3$$

$$\begin{aligned}X = X \cup \{x_1 = s_u\} &\cup \{x_2 = s_{k_u}\} \\ &= X \cup \{x_1 = s_3\} \cup \{x_2 = s_{k_3}\} \\ &= X \cup \{x_1 = s_3\} \cup \{x_2 = s_1\} \\ &= \{x_1 = rainy, x_2 = sunny\}\end{aligned}$$

At $t=3$, we have:

$$W_{2k_331} = W_{2131} = (b_{12})^2 a_{11}\pi_1 = 0.2^2 * 0.5 * 0.33$$
$$= 0.0066$$

$$W_{2k_332} = W_{2132} = (b_{22})^2 a_{12}\pi_2 = 0.25^2 * 0.25 * 0.33$$
$$= 0.00515625$$

$$W_{2k_333} = W_{2133} = (b_{32})^2 a_{13}\pi_3 = 0.1^2 * 0.25 * 0.33$$
$$= 0.000825$$

$$u = \text{argmax}_j\{W_{213j}\} = \text{argmax}_j\{W_{2131}, W_{2132}, W_{2133}\} = 1$$

$$\begin{aligned}X = X \cup \{x_3 = s_u\} &= X \cup \{x_3 = s_1\} \\ &= \{x_1 = rainy, x_2 = sunny, x_3 = sunny\}\end{aligned}$$

As a result, the optimal state sequence is $X = \{x_1=rainy, x_2=sunny, x_3=sunny\}$, which is the same to the one from individually optimal procedure (see table 4), Viterbi algorithm (see table 5), and normal longest-path algorithm (see table 6). The resulted sequence $X = \{x_1=rainy, x_2=sunny, x_3=sunny\}$ that is the longest path is drawn as bold line from node X_0 to node X_{13} to node X_{21} to node X_{31} inside the state transition graph, as seen in following fig. 6.
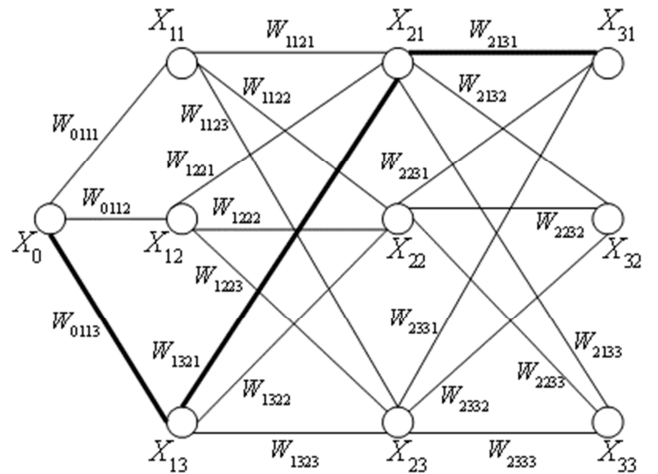

Figure 6. *Longest path drawn as bold line inside state transition graph.*

4. Conclusion

The longest-path algorithm proposes a new viewpoint in which uncovering problem is modeled as a graph. The different viewpoint is derived from the fact that longest-path algorithm keeps the optimal criterion as maximizing the con-

ditional probability $P(X|O, \Delta)$ whereas Viterbi algorithm maximizes the joint probability $P(X, O|\Delta)$. Moreover the longest-path algorithm does not use recurrence technique like Viterbi does but this is the reason that longest-path algorithm is less effective than Viterbi although the ideology of longest-path algorithm is simpler than Viterbi. It only moves forward and optimizes every 2-weight interval on the path. The way longest-path algorithm finds out longest path inside the graph shares the forward state transition with Viterbi algorithm. Therefore it is easy to recognize that the ideology of longest-path algorithm does not go beyond the ideology of Viterbi algorithm. However longest-path algorithm opens a potential research trend in improving solution of HMM uncovering problem when Viterbi algorithm is now the best algorithm with regard to theoretical methodology and we only enhance Viterbi by practical techniques. For example, authors [4] applied Hamming distance table into improving Viterbi. Authors [5] propose a fuzzy Viterbi search algorithm which is based on Choquet integrals and Sugeno fuzzy measures. Authors [6] extended Viterbi by using maximum likelihood estimate for the state sequence of a hidden Markov process. Authors [7] proposed an improved Viterbi algorithm based on second-order hidden Markov model for Chinese word segmentation. Authors [8] applied temporal abstraction into speeding up Viterbi. According to authors [9], the Viterbi can be enhanced by parallelization technique in order to take advantages of multiple CPU (s). According to authors [10], fangled decoder helps Viterbi algorithm to consume less memory with no error detection capability. They [10] also proposed a new efficient fangled decoder with less complexity which decreases significantly the processing time of Viterbi along with 2 bit error correction capabilities. Authors [11] combined posterior decoding algorithm and Viterbi algorithm in order to produce the posterior-Viterbi (PV). According to [11], "PV is a two step process: first the posterior probability of each state is computed and then the best posterior allowed path through the model is evaluated by a Viterbi algorithm". PV achieves strong points of both posterior decoding algorithm and Viterbi algorithm.

References

[1] J. G. Schmolze, "An Introduction to Hidden Markov Models," 2001.

[2] E. Fosler-Lussier, "Markov Models and Hidden Markov Models: A Brief Tutorial," 1998.

[3] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, no. 2, pp. 257-286, 1989.

[4] X. Luo, S. Li, B. Liu and F. Liu, "Improvement of the viterbi algorithm applied in the attacks on stream ciphers," in The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005, Dublin, 2005.

[5] N. P. Bidargaddi, M. Chetty and J. Kamruzzaman, "A Fuzzy Viterbi Algorithm for Improved Sequence Alignment and Searching of Proteins," in Applications of Evolutionary Computing, F. Rothlauf, J. Branke, S. Cagnoni, D. W. Corne, R. Drechsler, Y. Jin, P. Machado, E. Marchiori, J. Romero, G. D. Smith and G. Squillero, Eds., Lausanne, Springer Berlin Heidelberg, 2005, pp. 11-21.

[6] R. A. Soltan and M. Ahmadian, "Extended Viterbi Algorithm for Hidden Markov Process: A Transient/Steady Probabilities Approach," International Mathematical Forum, vol. 7, no. 58, pp. 2871-2883, 2012.

[7] L. La, Q. Guo, D. Yang and Q. Cao, "Improved Viterbi Algorithm-Based HMM2 for Chinese Words Segmentation," in The 2012 International Conference on Computer Science and Electronics Engineering, Hangzhou, 2012.

[8] S. Chatterjee and S. Russell, "A temporally abstracted Viterbi algorithm," arXiv.org, vol. 1202.3707, 14 February 2012.

[9] D. Golod and D. G. Brown, "A tutorial of techniques for improving standard Hidden Markov Model algorithms," Journal of Bioinformatics and Computational Biology, vol. 7, no. 04, pp. 737-754, August 2009.

[10] K. S. Arunlal and S. A. Hariprasad, "An Efficient Viterbi Decoder," International Journal of Computer Science, Engineering and Applications (IJCSEA), vol. 2, no. 1, pp. 95-110, February 2012.

[11] P. Fariselli and P. L. Martelli, "A new decoding algorithm for hidden Markov models improves the prediction of the topology of all-beta membrane proteins," BMC Bioinformatics, vol. 6(Suppl 4), no. S12, 1 December 2005.