Data Mining Lab, Big Data Research Center, UESTC
Email：junmshao@uestc.edu.cn
http://staff.uestc.edu.cn/shaojunming

# Something about Hidden Marcov Models

*Chongming Gao*

# Framework

**1. Markov Chains and Markov Property**

- Examples of Markov Chains
- Something about Markov Property

**2. Hidden Markov Models**

- Definition and Examples
- Three classic Problems
  - A. Evaluation Problem: Forward/Backward Algorithm
  - B. Decoding Problem: Viterbi Algorithm (Dynamic Programming)
  - C. Learning Problem: Baum-Welch Algorithm(EM Algorithm & GMM)

**3. Applications of HMM**

- Speech Recognition
- On-Line Hand Written Digits
- Computational Biology

**4. Other Issues of HMMs**

- Types of HMMs
- Implementation Issues

**5. Discussion: Generalize to Conditional Random Field**

Notation：

1. 本次组会PPT篇幅较长，可能非常耗时，对于没有接触过马尔科夫模型的同学来说，理解比较难（也不排除我把简单问题讲复杂的可能）。所以请保持注意力集中。

2. 对于已经了解HMM的同学，可以把重点放在我补充的一些干货上。

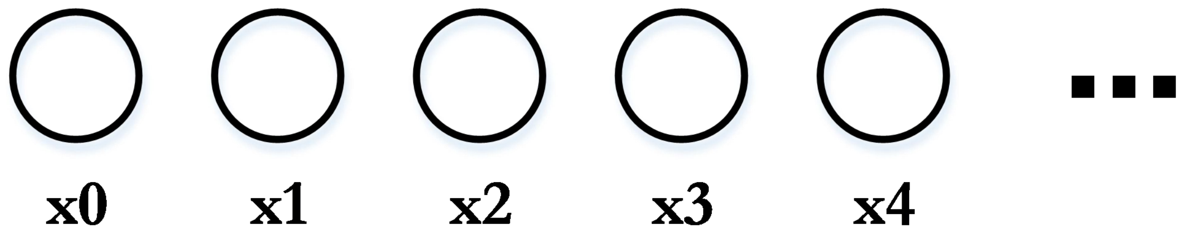
3.本PPT所有变量名都统一过，不会出现混淆的情况。

4. 若本次不能讲完，将分为(上、下)两次或抛弃后半部分。

# 1.1 Concepts About Markov Models

## ➢Random Process on Sequential Data

**Random process**, is a collection of random variables, representing the evolution of some system of random values over time. (Different from the i.i.d. variables)

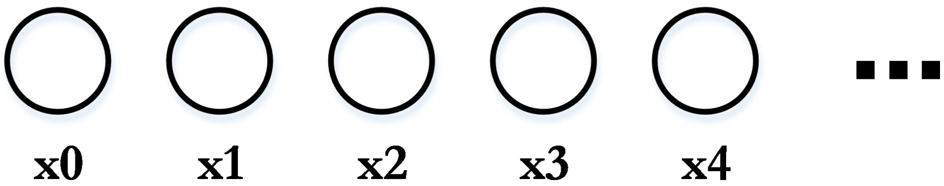



**A sequence of random variables**

# 1.1 Concepts About Markov Models

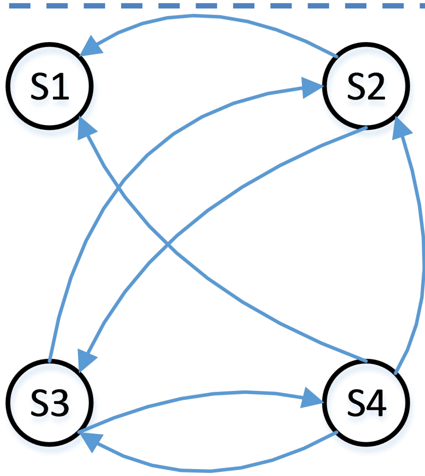
## ➢ Definition

**Markov chain**(discrete time) is a random process that undergoes transitions from one state to another on a state space.




$p(x1, x2, \cdots, xN)$ remains undefined

**A sequence of random variables**




**Transition diagram**

***Si*** is the value(state) of the discrete random variable

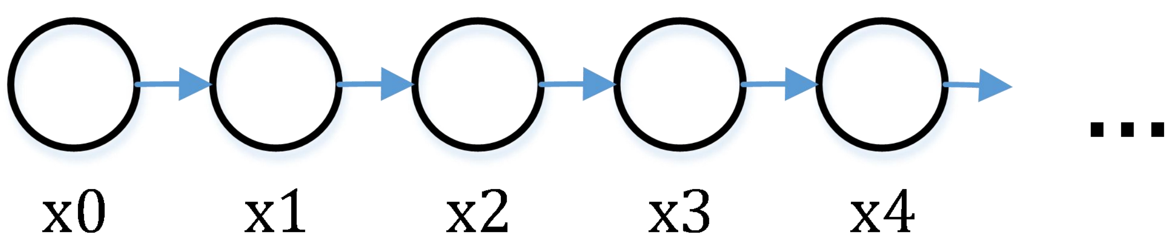
# 1.1 Concepts About Markov Models

- Markov Property:

The conditional probability distribution of future states of the process depends only upon the present state, not on the sequence of events that preceded it.

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = p(\mathbf{x}_1) \prod_{n=2}^{N} p(\mathbf{x}_n | \mathbf{x}_{n-1}).$$

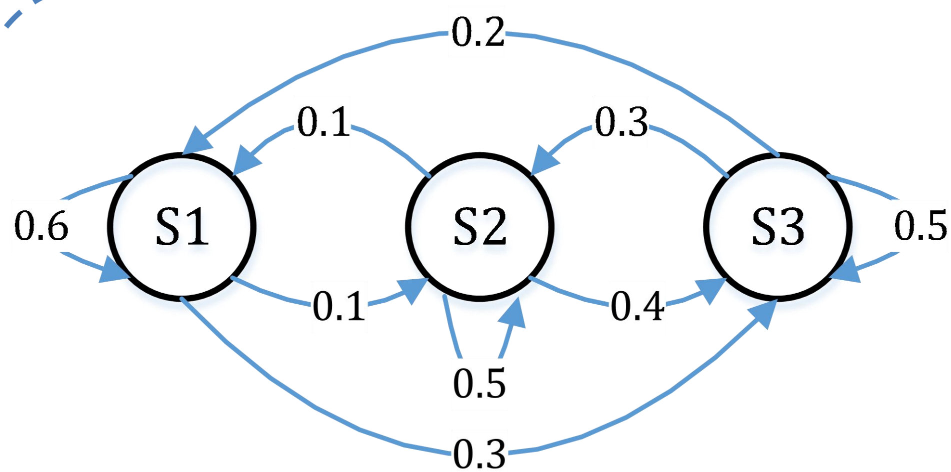
$$p(\mathbf{x}_n | \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) = p(\mathbf{x}_n | \mathbf{x}_{n-1})$$




**1-Order Markov Chains**

# 1.1 Concepts About Markov Models

## ➢1-Order Markov Model




**Transition diagram**

$$A = \begin{Bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{Bmatrix}$$

**Transition Matrix**

$$\pi = \begin{Bmatrix} 0.3 \\ 0.4 \\ 0.4 \end{Bmatrix}$$

**Initial Probability**

Model Parameters: $\lambda = (A, \pi)$

Note that transition diagram does not represent a probabilistic graphical model, because the nodes are not separate variables but rather states of a single variable

# 1.1 Concepts About Markov Models

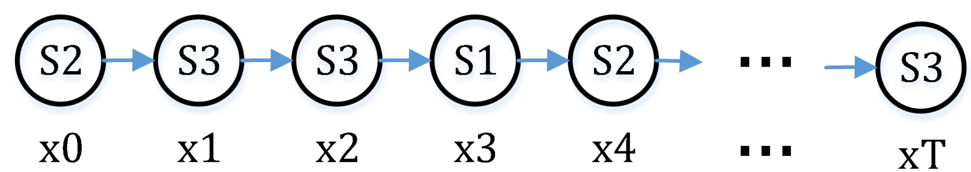
- How can we generate a sequence of observations?

Input: 1. Model $\lambda = (A, \pi)$
2. Sequence length $T$

$$A = \begin{Bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{Bmatrix} \qquad \pi = \begin{Bmatrix} 0.3 \\ 0.4 \\ 0.4 \end{Bmatrix}$$

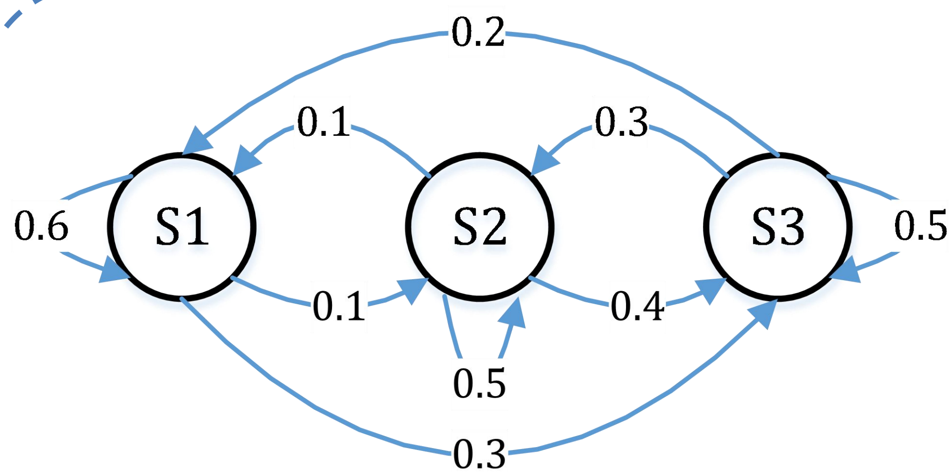
**Transition Matrix** **Initial Probability**

Output: A sequence of data
$O = (o_1, o_2, \dots o_T)$



# 1.1 Concepts About Markov Models

➢ Example: Weather Forecast, Stock Exchange




**Transition diagram**

$$A = \begin{Bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{Bmatrix}$$

**Transition Matrix**

$$\pi = \begin{Bmatrix} 0.3 \\ 0.4 \\ 0.4 \end{Bmatrix}$$

**Initial Probability**

Three states of weather on one day:

- S1: Sunny
- S2: Rainy
- S3: Cloudy

Three states of price of stock:

- S1: Stable
- S2: Increase
- S3: Decrease

# 1.1 Concepts About Markov Models

## ➢ Some Issues

1. Given the model parameters, as well as a sequence of variables represented the happened events, calculate the probability.

$$\underset{x0}{S2} \rightarrow \underset{x1}{S3} \rightarrow \underset{x2}{S3} \rightarrow \underset{x3}{S2} \rightarrow \underset{x4}{S1} \rightarrow \cdots$$

$$\pi = \left\{\begin{matrix} 0.3 \\ 0.4 \\ 0.4 \end{matrix}\right\} \quad A = \left\{\begin{matrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{matrix}\right\} \quad \Longrightarrow \quad p(x1, x2, \cdots, xN) = ?$$

2. Given a set of events happened and the initial distribution, estimate the parameters in the transition matrix

$$S1 \rightarrow S2 \rightarrow S1 \rightarrow S1 \rightarrow S3 \rightarrow$$

$$S2 \rightarrow S3 \rightarrow S3 \rightarrow S1 \rightarrow S2 \rightarrow$$

$$\vdots$$

$$S2 \rightarrow S3 \rightarrow S3 \rightarrow S1 \rightarrow S2 \rightarrow$$

$$\pi = \left\{\begin{matrix} 0.3 \\ 0.4 \\ 0.4 \end{matrix}\right\} \quad \Longrightarrow \quad A = \left\{\begin{matrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{matrix}\right\}$$

# 1.1 Concepts About Markov Models

## ➢Some Issues

3. Solve the convergence state. Suppose there exist a distribution

$$C = \{c1 \quad c2 \quad c3\}$$

in which the probability will keep stable after multiple by the transition matrix.

$$\{c1 \quad c2 \quad c3\} \cdot \begin{Bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{Bmatrix} = \{c1 \quad c2 \quad c3\}$$

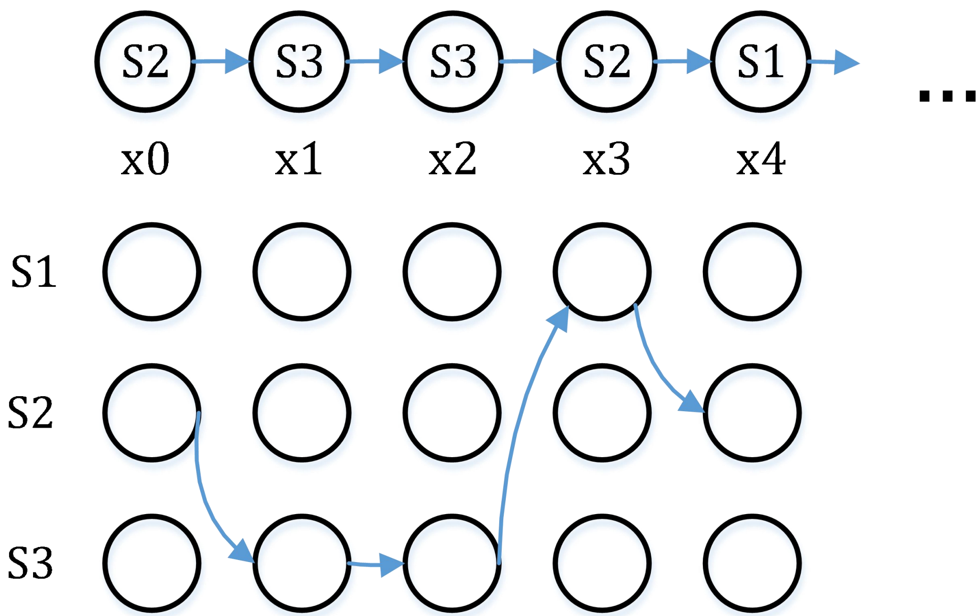
$$(C \cdot A = C)$$

$$\Rightarrow$$

$$C = \{c1 \quad c2 \quad c3\} = \{? \quad ? \quad ?\}$$

## ➢Some Issues

1. Given model parameters, calculate probability of a particular consequence variables.




$$A = \begin{Bmatrix} 0.6 & 0.1 & 0.3 \\ 0.1 & 0.5 & 0.4 \\ 0.2 & 0.3 & 0.5 \end{Bmatrix}$$

$$\pi = \begin{Bmatrix} 0.3 \\ 0.4 \\ 0.4 \end{Bmatrix}$$

$$P(x0 = S2, \cdots, x4 = S1) = \pi^T \cdot \begin{Bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{Bmatrix} \cdot a_{23} \cdot a_{33} \cdot a_{31} \cdot a_{12}$$

# 1.1 Concepts About Markov Models

## ➢Some Issues

2. Given a set of events happened and the initial distribution, estimate the parameters in the transition matrix

$S1 \rightarrow S2 \rightarrow S1 \rightarrow S1 \rightarrow S3 \rightarrow$

$S2 \rightarrow S3 \rightarrow S3 \rightarrow S1 \rightarrow S2 \rightarrow$

$\vdots$

$S2 \rightarrow S3 \rightarrow S3 \rightarrow S1 \rightarrow S2 \rightarrow$

$$\pi = \begin{Bmatrix} 0.3 \\ 0.4 \\ 0.4 \end{Bmatrix} \Longrightarrow A = \begin{Bmatrix} ? & ? & ? \\ ? & ? & ? \\ ? & ? & ? \end{Bmatrix}$$

$$a_{ij} = P\left(x_n = S_j \middle| x_{n-1} = S_i\right) = \frac{P(x_n = S_j, x_{n-1} = S_j)}{P(x_{n-1} = S_j)}$$

# 1.1 Concepts About Markov Models

## ➢Some Issues

3. Solve the convergence state. Suppose there exist a distribution

$$C = \{c1 \quad c2 \quad c3\}$$

in which the probability will keep stable after multiple by the transition matrix.

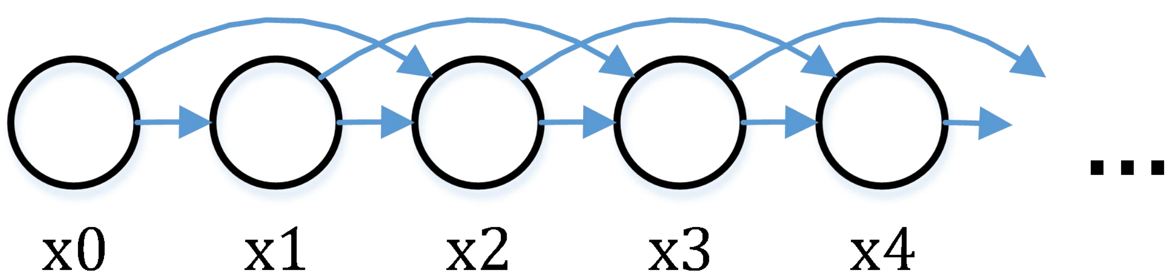
$$\begin{cases} C \cdot A = C \\ \sum_{i=1}^{|s|} ci = 1 \end{cases} \Longrightarrow \lim_{n \to \infty} C^n = \begin{Bmatrix} c1 & c2 & c3 \\ c1 & c2 & c3 \\ c1 & c2 & c3 \end{Bmatrix}$$

The mathematical description is similar to the fundamentals of algorithm PageRank, which will be introduced in *Xiaolin*'s presentation.

- **Extension: High Order Markov Chains**

The conditional probability distribution of future states of the process depends only *k* states of corresponding events.

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)\prod_{n=3}^{N} p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{x}_{n-2}).$$



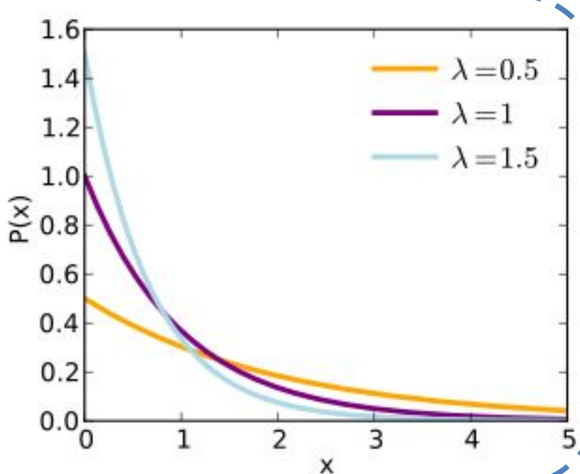

**2-Order Markov Chains**

# 1.2 Something about memorylessness(Generic)
(无后效性)

1. In probability and statistics, memorylessness is a property of certain probability distributions: the exponential distributions of non-negative real numbers and the geometric distributions of non-negative integers.

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \ge 0, \\ 0 & x < 0. \end{cases}$$

Exponential probability density




$$P(X > t+s|X > t) = \frac{P\{X > t, X > t+s\}}{P\{X > t\}} = \frac{P\{X > t+s\}}{P\{X > t\}}$$

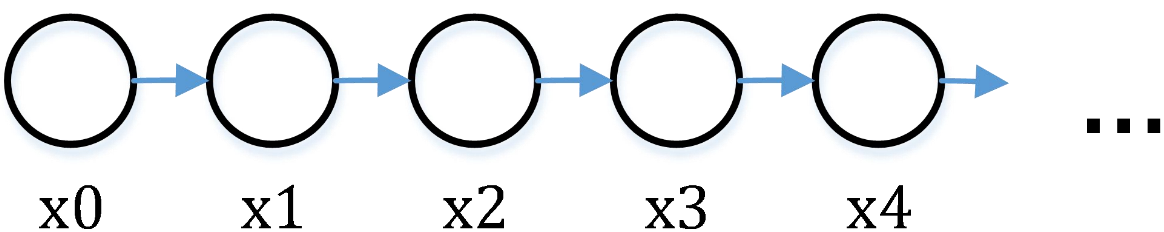
$$= \frac{e^{-\lambda(t+s)}}{e^{-\lambda t}} = e^{-\lambda s}$$

$$= P\{X > s\}$$

# 1.2 Something about memorylessness(Generic)
(无后效性)

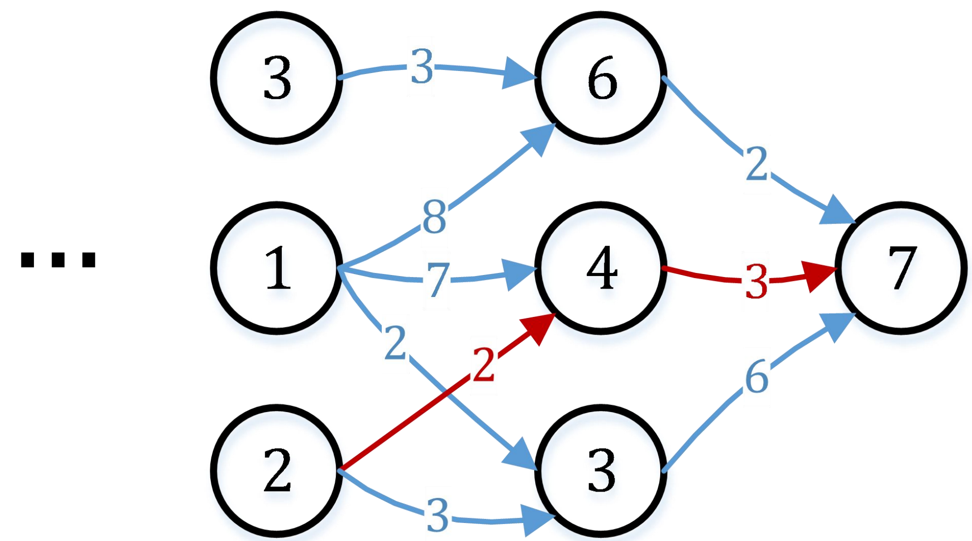
2. In Markov Model, "memorylessness" are used in a very different way, in which the underlying assumption of the Markov property implies that the properties of random variables related to the future depend only on relevant information about the current time, not on information from further in the past.

$$p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}) = p(\mathbf{x}_n|\mathbf{x}_{n-1})$$



# 1.2 Something about memorylessness(Generic)
(无后效性)

3. In dynamic programming, The memorylessness means the current optimal solution is strictly derived from the optimal solution from last state, while not considering how did the previous optimal solution come into being.




**Search for the minimum cost of a sequence**

*(The essence of dynamic programming will be discuss later in this slides)*

# Framework

**1. Markov Chains and Markov Property**

- Examples of Markov Chains
- Something about Markov Property

**2. Hidden Markov Models**

- Definition and Examples
- Three classic Problems
  - A. Evaluation Problem: Forward/Backward Algorithm
  - B. Decoding Problem: Viterbi Algorithm (Dynamic Programming)
  - C. Learning Problem: Baum-Welch Algorithm(EM Algorithm & GMM)

**3. Applications of HMM**

- Speech Recognition
- On-Line Hand Written Digits
- Computational Biology

**4. Other Issues of HMMs**

- Types of HMMs
- Implementation Issues

**5. Discussion: Generalize to Conditional Random Field**

# 2.1 Hidden Markov Models

- Motivation：Professor Shao's Procrastination(拖延症) in Driving Practice

As we all know, our dear God Professor Shao, is a prominent, excellent, extraordinary, marvelous and distinguished scholar.

He hates delay and procrastination in all its form, except on driving practice

# 2.1 Hidden Markov Models

- Activation: Professor Shao's Procrastination(拖延症) in Driving Practice

In our world, Professor Shao has only 2 states on one day, that is going to drive car or not going to drive car.

0 → 0 → 0 → 1 → 0 → … → 0 →

0: No, he doesn't drive
1: Yes, he drive! Unbelievable!

However, we don't know what exactly in Shao's mind. Actually, in his mind there are 4 states:

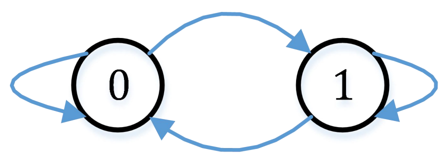
N → L → H → C → N → N → H →

N: Normal
L: Light
H: Heavy
C: Critical

# 2.1 Hidden Markov Models

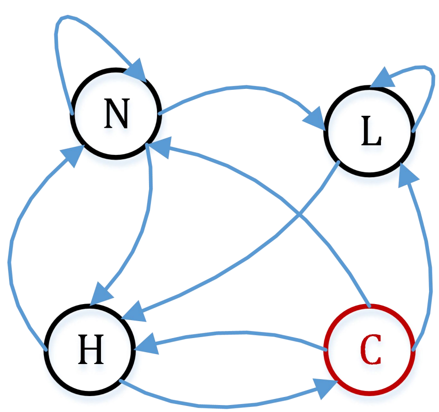
- Motivation： Professor Shao's Procrastination(拖延症)
  in Driving Practice

Traditional Markov Model fails to describe this probability transition, since there are something we can't directly observe! However, there is a probability pattern can be describe with a latent variable latent variable(the state of Shao's mind)




0: No, he doesn't drive
1: Yes, he drive! Unbelievable!

**Traditional Markov Model**



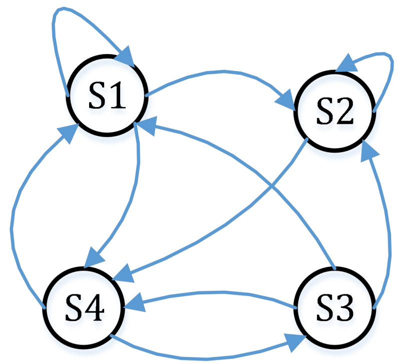

N: Normal
L: Light
H: Heavy
C: Critical

**Transition diagram of the latent variable**

# 2.1 Hidden Markov Models

➢ HMM Model

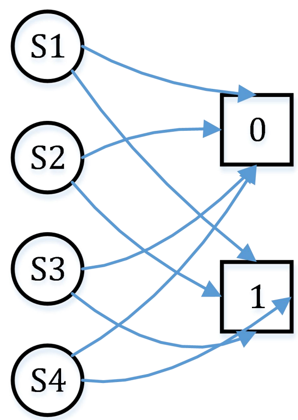
$$\lambda = (A, B, \pi)$$




Transition diagram

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}$$

Transition matrix




Emission diagram

$$B = \begin{pmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{pmatrix}$$

Emission matrix

Initial State

$$\pi = (p_1 \quad \cdots \quad p_n)$$

# 2.1 Hidden Markov Models

➢ How can we generate a sequence of observations?

Input: 1. Model $\lambda = (A, B, \pi)$

2. Sequence length $T$

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \mathrm{B} = \begin{pmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{pmatrix}$$

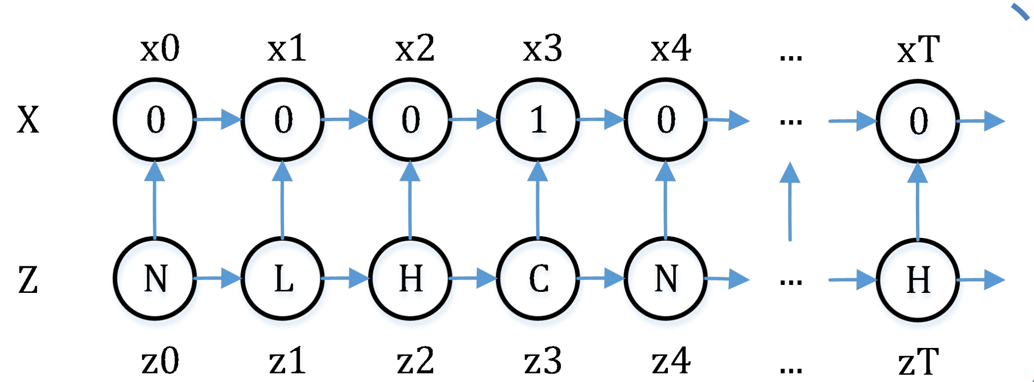
$$\pi = (p_1 \quad \cdots \quad p_n)$$

Output: Sequences of observe data:

$$X = (x_1, x_2, \dots x_T, )$$

Latent data:

$$Z = (z_1, z_2, \dots z_T, )$$



# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

  Given the observation sequence $X = (x1, x2, \ldots xT)$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(X \mid \lambda)$?

  - Forward/Backward Algorithm

- 2. **Decoding Problem**(Predicting Problem):

  Given the a model $\lambda = (A, B, \pi)$, how do we choose a corresponding state sequence $X = (x_1, x_2, \ldots x_T,)$ which is optimal in some meaningful sense?

  - Viterbi Algorithm
  - (Dynamic Programming)

- 3. **Learning Problem**(Training Problem):

  Given a set of observation sequences $X = (x1, x2, \ldots xT)$, How do we adjust the model parameters to maximize $P(X \mid \lambda)$?

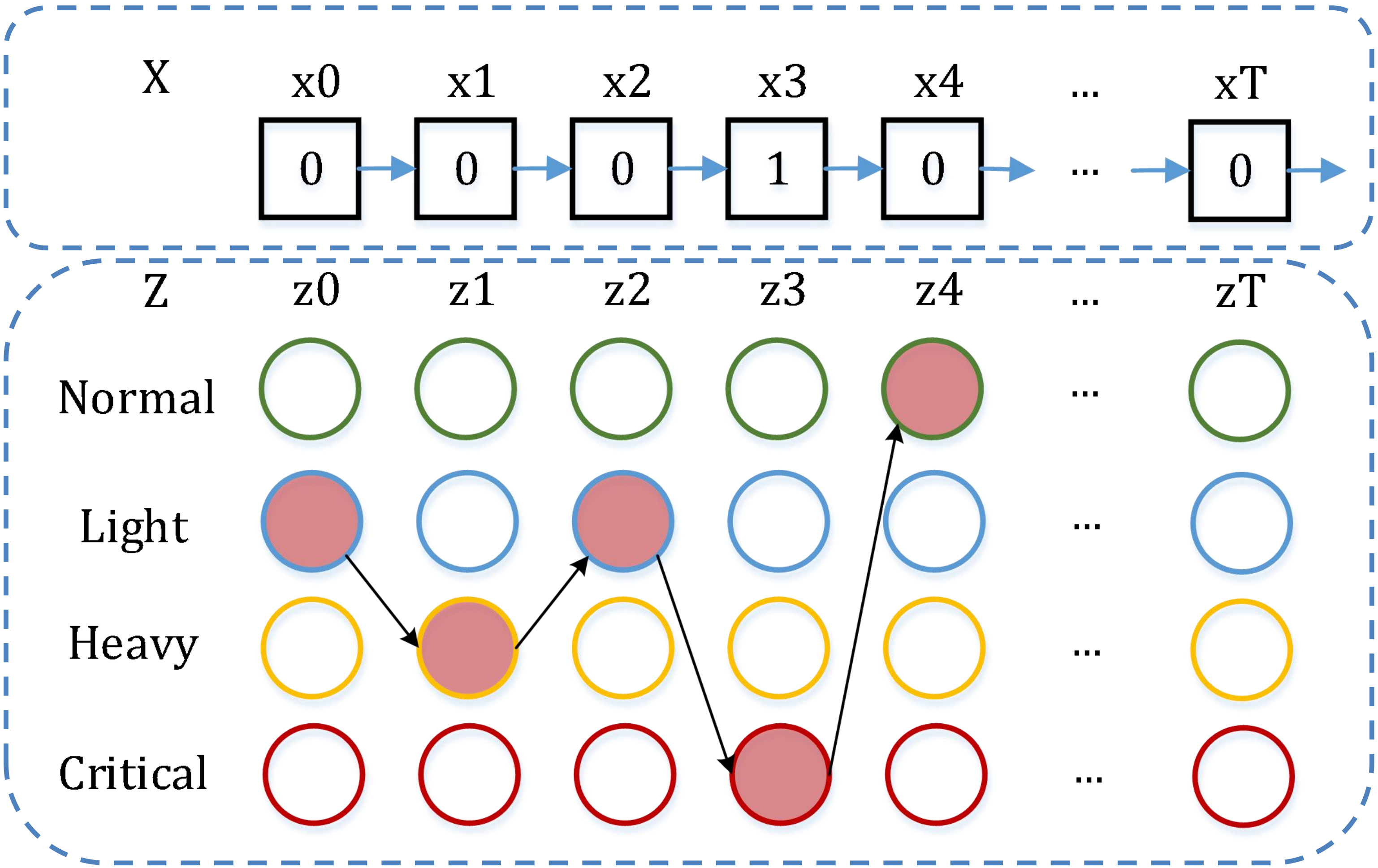
  - Baum-Welch Algorithm
  - (EM Algorithm & GMM)

用上下页的图分别说

# 2.2 Three Classic Problems

➢ 1. **Evaluation Problem** (Calculation Problem):

Given the observation sequence $X = (x1, x2, \ldots xT)$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(X \mid \lambda)$?



# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Method 1: Directly Computing:**

Given the observation sequence $X = (x1, x2, \dots xT)$, for a special latent sequence $Z = (z1, z2, \dots zT)$

$$P(Z|\lambda) = \pi_{z1} \cdot a_{z1,z2} \cdot a_{z2,z3} \cdot a_{z3,z4} \cdots a_{Z_{T-1},Z_T}$$

The condition probability is computing as:

$$P(X\,|Z,\lambda) = b_{z1}(x1) \cdot b_{z2}(x2) \cdots b_{zT}(xT)$$

Final result is:

$$\boxed{\begin{aligned} P(X \mid \lambda) &= \sum_{Z} P(X\,|Z,\lambda)P(Z|\lambda) \\ &= \sum_{Z} \pi_{z1} a_{z1,z2}\, b_{z1}(x1) a_{z2,z3} b_{z2}(x2) \cdots a_{Z_{T-1},Z_T} b_{zT}(xT) \end{aligned}}$$

Unfortunately, This calculation is computationally unfeasible, even for small values of $n$ and $T$; e.g., for N = 4 (states), T = 100(observations), there are on the order of $2 \cdot 100 \cdot 4^{100} \approx 10^{62}$ computations!

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Method 2: Forward Algorithm**

Consider the forward variable $\alpha_t(i)$ defined as:

$$\alpha_t(i) = P(x_1, x_2, \cdots, x_t, z_t = S_i \mid \lambda)$$

i.e., the probability of the partial observation sequence, $x_1, x_2, \cdots, x_t$, (until time $t$) and state $S_i$, at time $t$, given the model $\lambda$. We can solve for $\alpha_t(i)$ inductively, as follows:

1) Initialization:

$$\alpha_1(i) = \pi_i b_i(x1), \qquad 1 \le i \le N$$

2) Induction:

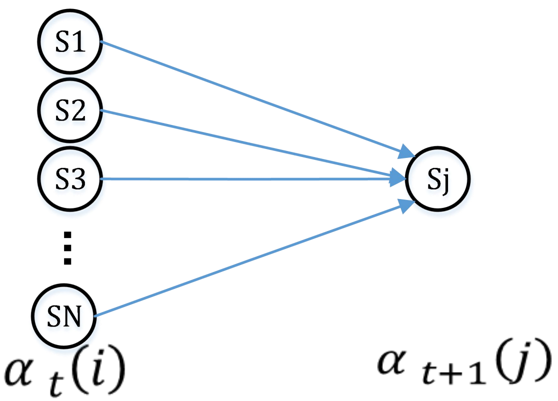
$$\alpha_{t+1}(j) = [\textstyle\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_i(x_{t+1}), \qquad 1 \le t \le T-1, 1 \le j \le N$$

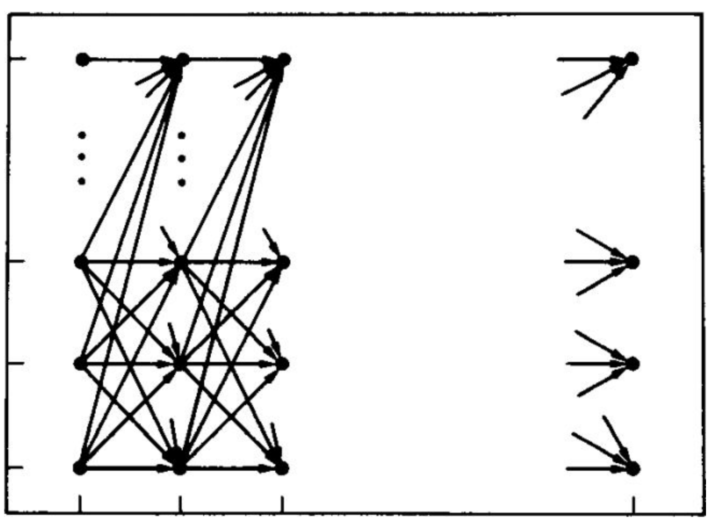
3) Termination:

$$P(X \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

- 1. **Evaluation Problem** (Calculation Problem):

**Method 2: Forward Algorithm**



$$\alpha_t(i) = P(x_1, x_2, \cdots, x_t, z_t = S_i \mid \lambda)$$

1) Initialization:

$$\alpha_1(i) = \pi_i b_i(x1), \qquad 1 \le i \le N$$

2) Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_i(x_{t+1})$$

$$1 \le t \le T-1, 1 \le j \le N$$

3) Termination:

$$P(X \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Method 3: Backward Algorithm**

Consider the forward variable $\beta_t(i)$ defined as:

$$\beta_t(i) = P(x_{t+1}, x_{t+2}, \cdots, x_T, z_t = S_i \mid \lambda)$$

i.e., the probability of the partial observation sequence, $x_1, x_2, \cdots, x_t$, (until time $t$) and state $S_i$, at time $t$, given the model $\lambda$. We can solve for $\beta_t(i)$ inductively, as follows:

1) Initialization:

$$\beta_T(i) = 1, \qquad 1 \le i \le N$$

2) Induction:

$$\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(x_{t+1})\right] \beta_{t+1}(j), \quad t = T-1, T-2, \cdots, 1, 1 \le i \le N$$

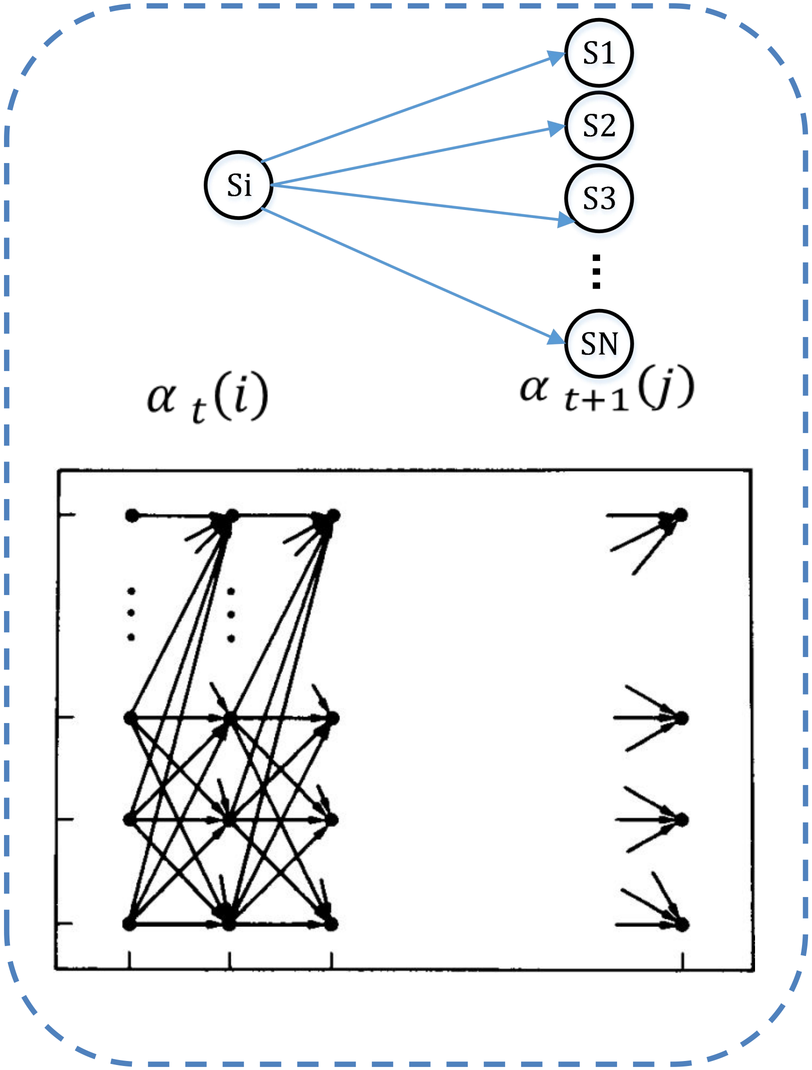
3) Termination:

$$P(X \mid \lambda) = \sum_{i=1}^{N} \pi_i b_i(x1) \beta_1(i)$$

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Method 3: Backward Algorithm**




$$\beta_t(i) = P(x_{t+1}, x_{t+2}, \cdots, x_T, z_t = S_i \mid \lambda)$$

1) Initialization:

$$\beta_T(i) = 1, \qquad 1 \le i \le N$$

2) Induction:

$$\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(x_{t+1})\right] \beta_{t+1}(j)$$

$$t = T-1, T-2, \cdots, 1, 1 \le i \le N$$

3) Termination:

$$P(X \mid \lambda) = \sum_{i=1}^{N} \pi_i b_i(x1) \beta_1(i)$$

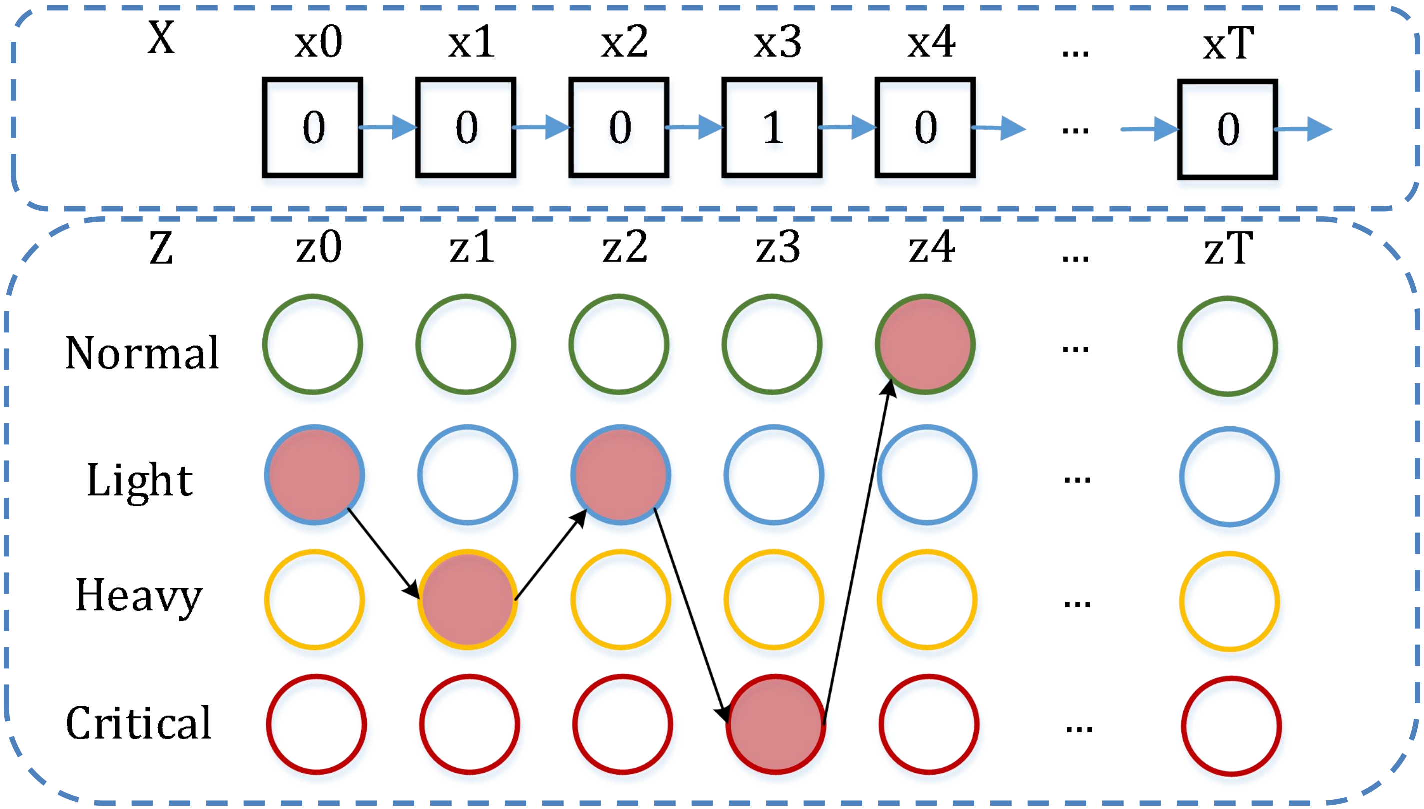
# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Discussion:**

The complexity of Direct Computing is $O(TN^T)$, The complexity of Forward/Backward Algorithm is $O(TN^2)$.

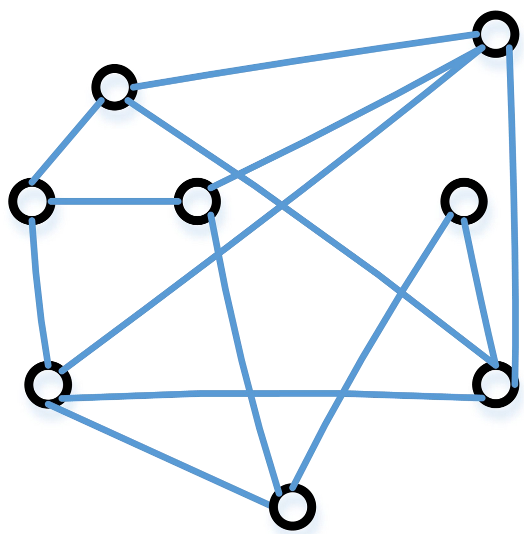
Why? Direct Computing has a lot of redundant computations!



# 2.2 Three Classic Problems

➢ 1. **Evaluation Problem** (Calculation Problem):

Extension: Floyd Algorithm, computed the shortest path of all pairs on a net , whose complexity is $O(N^3)$.



**A network**

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Some interesting probability calculations:**

1) define the variable:

$$\gamma_t(i) = P(z_t = S_i | X, \lambda)$$

i.e., the probability of being in state $S_i$ at time $t$, given the observation sequence $X$, and the model $\lambda$.

$$\gamma_t(i) = P(z_t = S_i | X, \lambda) = \frac{P(z_t = S_i, X | \lambda)}{P(X|\lambda)}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{P(X|\lambda)}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

**Some interesting probability calculations:**

2) define the variable:

$$\xi_t(i,j) = P\left(z_t = S_i, z_{t+1} = S_j \middle| X, \lambda\right)$$

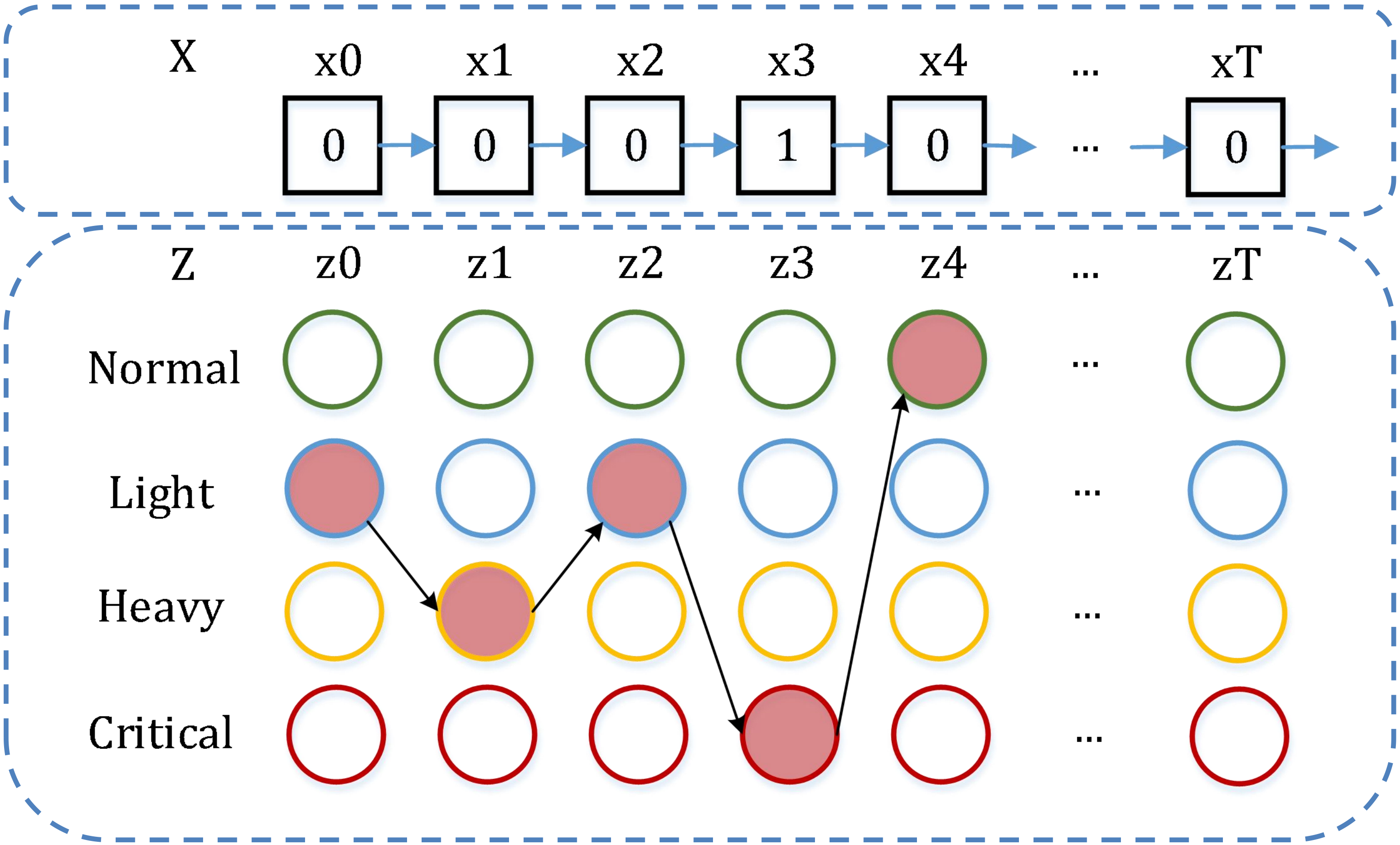
i.e., the probability of being in state $S_i$ at time $t$, and in state $S_j$ at time $t+1$ given the observation sequence $X$, and the model $\lambda$.

$$\begin{aligned}
\xi_t(i,j) &= P\left(z_t = S_i, z_{t+1} = S_j \middle| X, \lambda\right) \\
&= \frac{P\left(z_t = S_i, z_{t+1} = S_j, X \middle| \lambda\right)}{P(X|\lambda)} \\
&= \frac{P\left(z_t = S_i, z_{t+1} = S_j, X \middle| \lambda\right)}{\sum_{i=1}^{N} \sum_{i=1}^{N} P\left(z_t = S_i, z_{t+1} = S_j, X \middle| \lambda\right)} \\
&= \frac{\alpha_{\ t}(i) a_{ij} b_i(x_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{i=1}^{N} \alpha_{\ t}(i) a_{ij} b_i(x_{t+1}) \beta_{t+1}(j)}
\end{aligned}$$

# 2.2 Three Classic Problems

➢ 2. **Decoding Problem**(Predicting Problem):

Given the a model $\lambda = (A, B, \pi)$, how do we choose a corresponding state sequence $X = (x_1, x_2, \dots x_T,)$ which is optimal in some meaningful sense?



# 2.2 Three Classic Problems

- 2. **Decoding Problem**(Predicting Problem):

**Viterbi algorithm**

we need to define the quantity:

$$\delta_t(i) = \max_{z_1, z_2, \cdots, z_{T-1}} P(z_t = S_i, z_{t-1}, \cdots, z_1, x_t, x_{t-1}, \cdots, x_1, |\lambda)$$

i.e. The maximal probability of being in state $S_i$ at time $t$, given the observation sequence $X$, and the model $\lambda$.

Induction:

$$\begin{aligned}\delta_{t+1}(i) &= \max_{z_1, z_2, \cdots, z_{T-1}} P(z_{t+1} = S_i, z_{t-1}, \cdots, z_1, x_{t+1}, x_t, \cdots, x_1, |\lambda) \\ &= \max_{z_1, z_2, \cdots, z_{T-1}} [\delta_t(j) a_{ji}] b_i(x_{t+1})\end{aligned}$$

To find the sequence of latent variable values that corresponds to this path, defined:

$$\psi_t(i) = \arg\max_{z_1, z_2, \cdots, z_{T-1}} \delta_{t-1}(i) a_{ji}$$

i.e. The state of the $(t-1)$ th variable, when the $(t)$ come to the maximum

# 2.2 Three Classic Problems

- 2. **Decoding Problem**(Predicting Problem):

**Viterbi algorithm**

Input: Model parameter $\lambda = (A, B, \pi)$, observation sequence $X = (x1, x2, \dots xT)$

Output: Optimal latent sequence $Z = (z1^*, z2^*, \dots zT^*)$

1) Initialization:

$$\delta_1(i) = \pi_i b_i(x1), \qquad 1 \le i \le N$$

$$\psi_1(i) = 0, \qquad 1 \le i \le N$$

2) Induction:

$$\delta_{t+1}(i) = \max_{z_1, z_2, \cdots, z_{T-1}} [\delta_t(j) a_{ji}] b_i(x_{t+1}) \quad 1 \le i \le N$$

$$\psi_t(i) = \arg\max_{z_1, z_2, \cdots, z_{T-1}} \delta_{t-1}(i) a_{ji}, \qquad 1 \le i \le N$$

3) Termination:

$$P^* = \max_{1 \le i \le N} \delta_T(i)$$

$$z_T{}^* = \arg\max_{1 \le i \le N} \delta_T(i)$$

4) Optical path backtracking:

For $t = T-1, T-2, \dots, 1$

$$z_t{}^* = \psi_{t+1}(z_{t+1}{}^*)$$

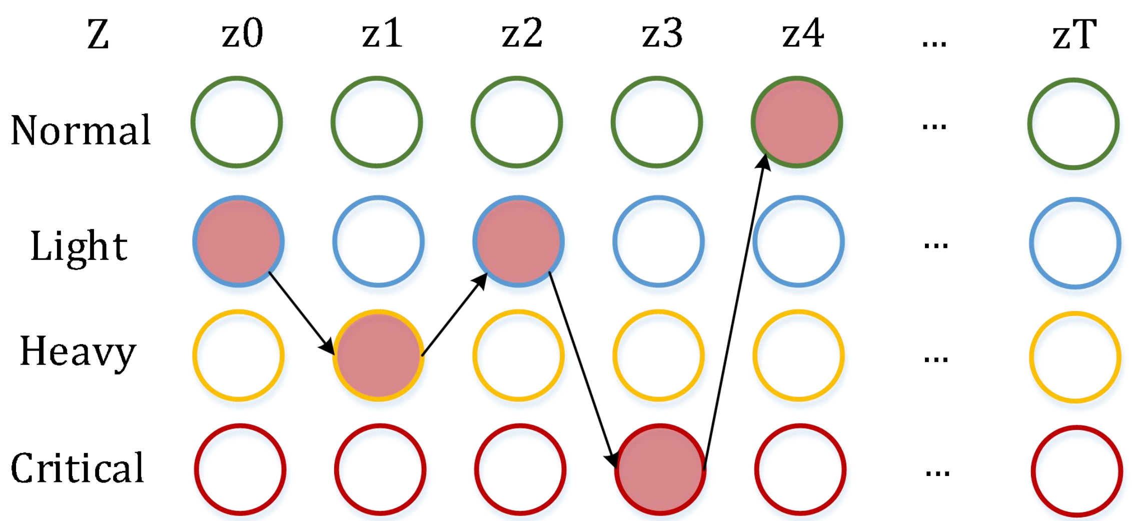
# 2.2 Three Classic Problems

- 2. **Decoding Problem**(Predicting Problem):

**Viterbi algorithm**

Input: Model parameter $\lambda = (A, B, \pi)$, observation sequence $X = (x1, x2, \ldots xT)$

Output: Optimal latent sequence $Z = (z1^*, z2^*, \ldots zT^*)$




every circle is a state, corresponding an entry in a 2-dimensional matrix. The $\delta_{t+1}(i)$ and $\psi_t(i)$ should be calculated on each entry of this matrix!

$$\delta_{t+1}(i) = \max_{z_1, z_2, \cdots, z_{T-1}} P(z_{t+1} = S_i, z_{t-1}, \cdots, z_1, x_{t+1}, x_t, \cdots, x_1, |\lambda)$$
$$= \max_{z_1, z_2, \cdots, z_{T-1}} [\delta_t(j) a_{ji}] b_i(x_{t+1})$$

$$\psi_t(i) = \arg\max_{z_1, z_2, \cdots, z_{T-1}} \delta_{t-1}(i) a_{ji}$$

# 2.2 Three Classic Problems

- The essence of Dynamic Programing

Dynamic programming is a method for solving a complex problem by breaking it down into a collection of simpler sub-problems.

**如何拆分问题**，才是动态规划的核心。
而**拆分问题**，靠的就是**状态的定义**和**状态转移方程的定义**。

**几种算法的实质[1]：**

**递推** -> 每个阶段只有一个状态；
**贪心** -> 每个阶段的最优状态都是由上一个阶段的最优状态得到的；
**搜索** -> 每个阶段的最优状态是由之前所有阶段的状态的组合得到的；
**动态规划** -> 每个阶段的最优状态可以从之前某个阶段的某个或某些状态直接得到而不管之前这个状态是如何得到的。(无后效性)

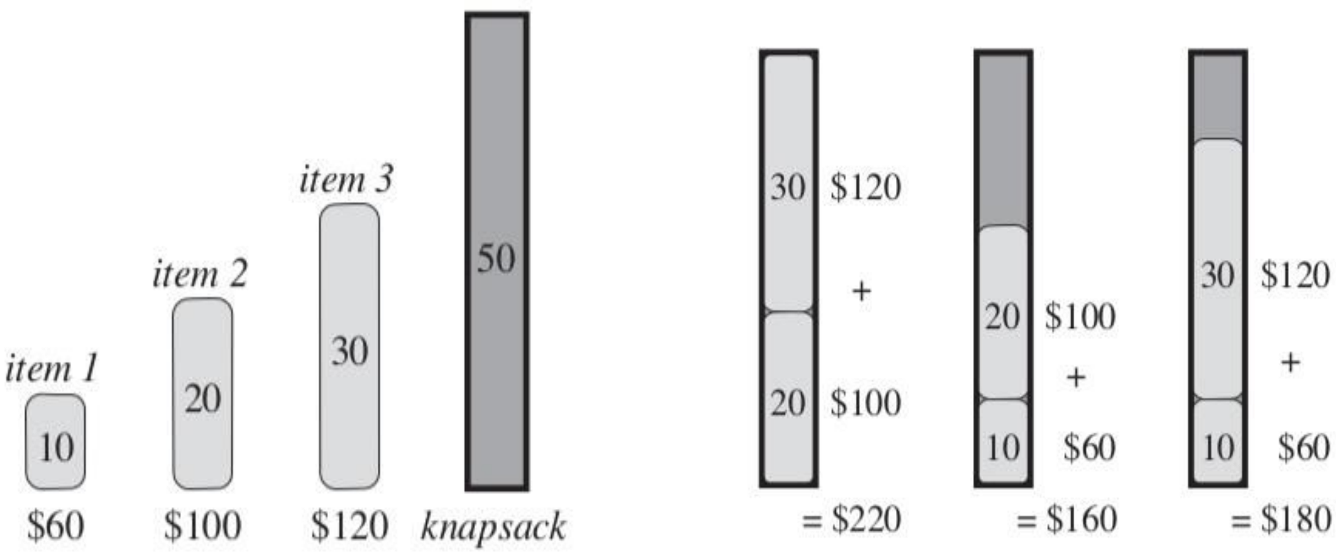
[1]*Site from: http://www.zhihu.com/question/23995189*

# 2.2 Three Classic Problems

➢ knapsack problem(背包问题)

问题描述：

有$N$件物品和一个容量为$V$的背包。放入第$i$件物品耗费的空间是$Ci$，得到的价值是$Wi$。求解将哪些物品装入背包可使价值总和最大。




# 2.2 Three Classic Problems

- knapsack problem(背包问题)

  - 0-1背包：每种物品仅有一件，可以选择放或不放。
  - 用子问题定义状态：即$F[i,v]$表示前$i$件物品恰放入一个容量为$v$的背包可以获得的最大价值。则其状态转移方程便是：

$$F[i,v]=max\{F[i-1,v],F[i-1,v-Ci]+Wi\}$$

1. 将前$i$件物品放入容量为$v$的背包中”这个子问题，若只考虑第$i$件物品的策略（放或不放），那么就可以转化为一个只和前$i-1$件物品相关的问题。

2. 如果不放第$i$件物品，那么问题就转化为“前$i-1$件物品放入容量为$v$的背包中”，价值为$F[i-1,v]$；

3. 如果放第$i$件物品，那么问题就转化为“前$i-1$件物品放入剩下的容量为$v-Ci$的背包中”，此时能获得的最大价值就是$F[i-1,v-Ci]$再加上通过放入第$i$件物品获得的价值$Wi$

# 2.2 Three Classic Problems

- knapsack problem(背包问题)

伪代码如下：

$$
\begin{aligned}
&F[0, 0..V] = 0\\
&\text{for } i = 1 \text{ to } N\\
&\quad \text{for } v = C_i \text{ to } V\\
&\qquad F[i, v] = \max\{F[i-1, v], F[i-1, v-C_i] + W_i\}
\end{aligned}
$$

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 2 | 0 | 0 | 0 | 2 | 2 | 3 | 3 | 3 | 5 | 5 | 5 |
| 3 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 6 | 6 | 7 | 7 |

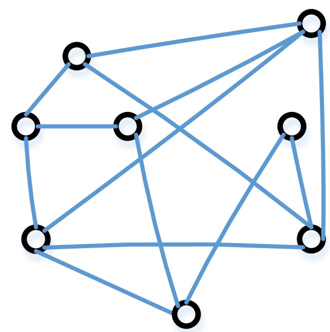
# 2.2 Three Classic Problems

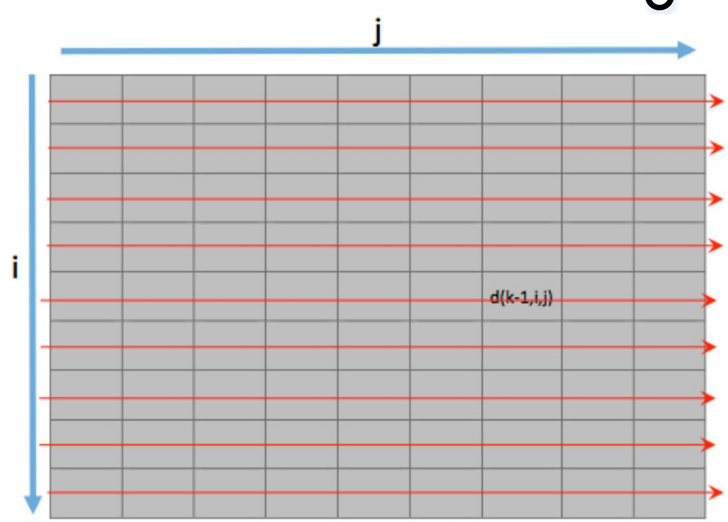
- Dynamic Programing in Floyd-Warshall Algorithm

Define $d[k][i][j]$ as the shortest path between node $i$ and node $j$, constricted by passed medial nodes $1\sim k$.

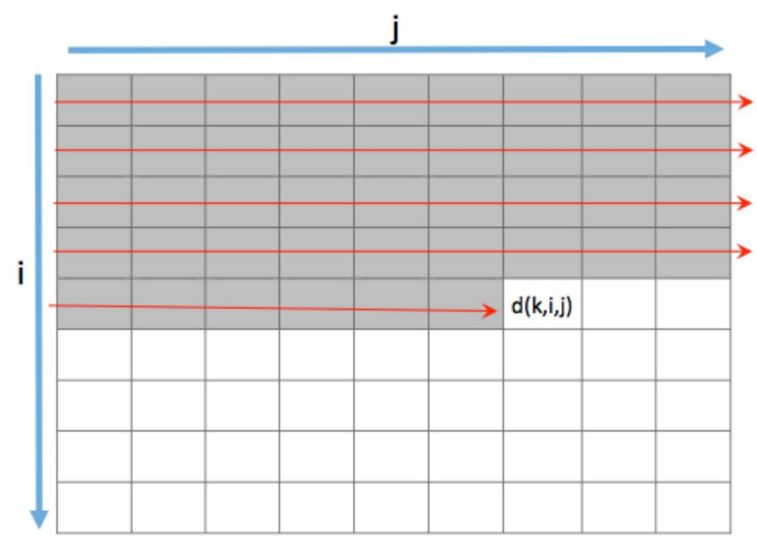
$$d[k][i][j] = \min(d[k-1][i][j], d[k-1][i][k] + d[k-1][k][j])$$

$$(k, i, j \in [1, n])$$






d[k-1][][]（第 k-1 阶段）

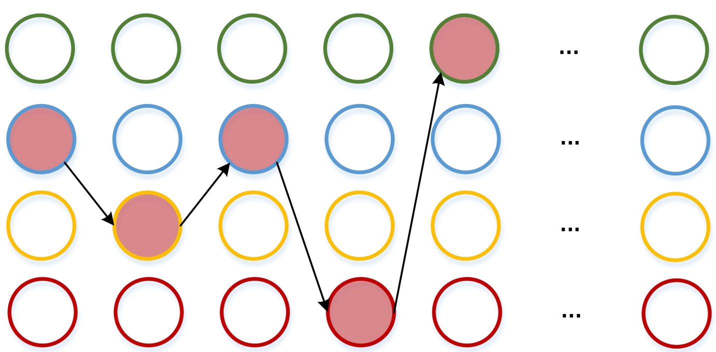


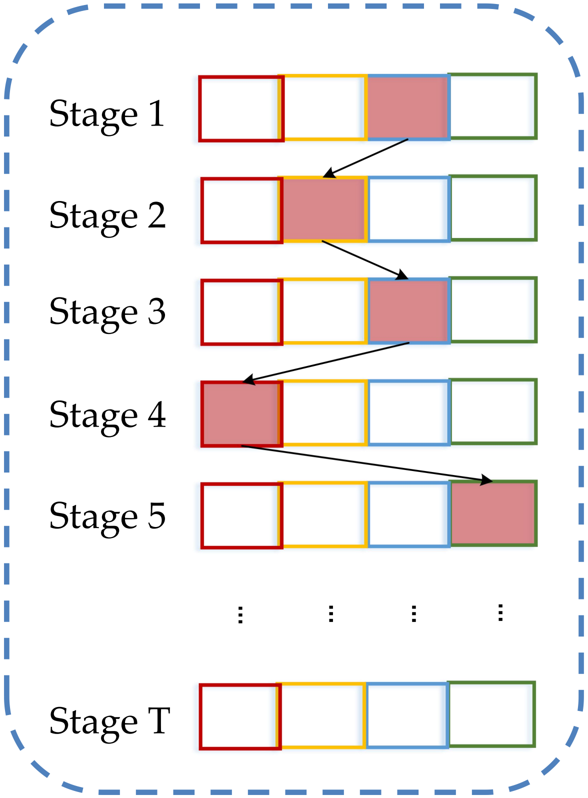

d[k][][]（第 k 阶段）

# 2.2 Three Classic Problems

➢ 2. **Decoding Problem**(Predicting Problem):

With Viterbi algorithm, now you can predict what's in Professor Shao's mind with a maximal probability !






Transition with dynamic programming thought

# 2.2 Three Classic Problems

- 1. **Evaluation Problem** (Calculation Problem):

  Given the observation sequence $X = (x1, x2, \ldots xT)$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(X \mid \lambda)$?

  - Forward/Backward Algorithm

- 2. **Decoding Problem**(Predicting Problem):

  Given the a model $\lambda = (A, B, \pi)$, how do we choose a corresponding state sequence $X = (x_1, x_2, \ldots x_T,)$ which is optimal in some meaningful sense?

  - Viterbi Algorithm
  - (Dynamic Programming)

- 3. **Learning Problem**(Training Problem):

  Given a set of observation sequences $X = (x1, x2, \ldots xT)$, How do we adjust the model parameters to maximize $P(X \mid \lambda)$?

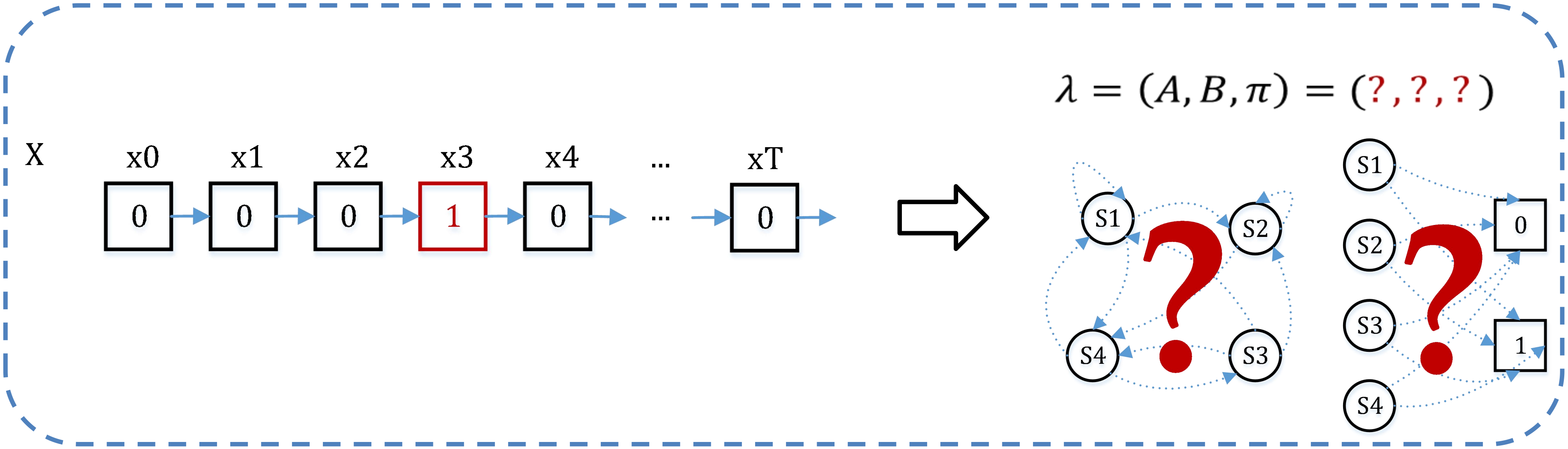
  - Baum-Welch Algorithm
  - (EM Algorithm & GMM)

用上下页的图分别说

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

Given a set of observation sequences $X = (x1, x2, \dots xT)$, How do we adjust the model parameters to maximize $P(X \mid \lambda)$?




The idea of estimating parameters: **Maximum likelihood estimation(MLE)**

$$\hat{\lambda} = \arg\max_{\lambda} logP(X \mid \lambda)$$

Due to the latent parameters $Z = (z1, z2, \dots zT)$, there are not analytic solutions. The numerical solutions can be derived iteratively.

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

**EM Algorithm**

Expectation–maximization (EM) algorithm is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables.

- ➢ Initialize parameters
- ➢ while not converged
  - **Expectation step:** Calculate log likelihood of the new parameters.

    严格来说是计算$Q(\theta,\theta^{(i)})$函数, 但其实质为用迭代过程上一次的参数$\theta^{(i)}$来计算一些概率，这些计算结果将在下一次**M**过程中用来估计新的参数$\theta$.
  - **Maximization step:** Estimate new parameters

    利用**E**步所得到的某些概率，估算新的参数$\theta$.

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

**EM Algorithm**

Take Gaussian mixture model(GMM) for example:

混合高斯分布的概率密度函数：

$$P(y|\theta)=\sum_{k=1}^{K}\alpha_k\phi(y|\theta_k)$$

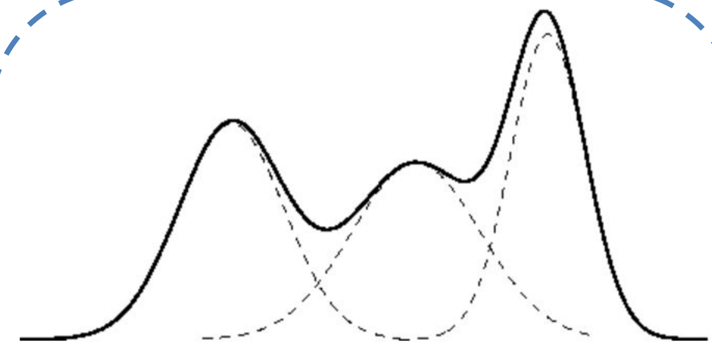
$$\phi(y|\theta_k)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp\left(-\frac{(y-\mu_k)^2}{2\sigma_k^2}\right)$$

需要估计的参数：每个高斯函数的权重 $\alpha_k$，以及每个高斯函数的均值、方差（协方差）$\theta_k=(\mu_k,\sigma_k)$

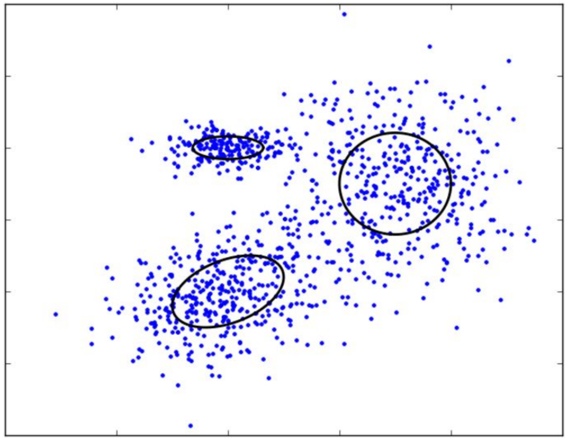
明确隐变量，即给一组样本点 $Y=(y1,y2,\ldots yN)$，第 $j$ 个点来自于第 $k$ 个模型，用 $\gamma_{jk}$ 表示：

$$\gamma_{jk}=\begin{cases}1, & \text{第 } j \text{ 个观测来自第 } k \text{ 个分模型} \\ 0, & \text{否则}\end{cases}$$

$$j=1,2,\cdots,N;\quad k=1,2,\cdots,K$$



1-Order GMM



2-Order GMM

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

**EM Algorithm on GMM**

**Expectation step**

$$Q(\theta,\theta^{(i)}) = E[\log P(y,\gamma|\theta)\,|\,y,\theta^{(i)}]$$

$$= E\left\{\sum_{k=1}^{K} n_k \log\alpha_k + \sum_{j=1}^{N}\gamma_{jk}\left[\log\left(\frac{1}{\sqrt{2\pi}}\right) - \log\sigma_k - \frac{1}{2\sigma_k^2}(y_j-\mu_k)^2\right]\right\}$$

$$= \sum_{k=1}^{K}\left\{\sum_{j=1}^{N}(E\gamma_{jk})\log\alpha_k + \sum_{j=1}^{N}(E\gamma_{jk})\left[\log\left(\frac{1}{\sqrt{2\pi}}\right) - \log\sigma_k - \frac{1}{2\sigma_k^2}(y_j-\mu_k)^2\right]\right\}$$

这里需要计算 $E(\gamma_{jk}|y,\theta)$，记为 $\hat{\gamma}_{jk}$ .

$$\hat{\gamma}_{jk} = E(\gamma_{jk}|y,\theta) = P(\gamma_{jk}=1|y,\theta)$$

$$= \frac{P(\gamma_{jk}=1, y_j|\theta)}{\sum_{k=1}^{K} P(\gamma_{jk}=1, y_j|\theta)}$$

$$= \frac{P(y_j|\gamma_{jk}=1,\theta)P(\gamma_{jk}=1|\theta)}{\sum_{k=1}^{K} P(y_j|\gamma_{jk}=1,\theta)P(\gamma_{jk}=1|\theta)}$$

$$= \frac{\alpha_k\phi(y_j|\theta_k)}{\sum_{k=1}^{K}\alpha_k\phi(y_j|\theta_k)}, \quad j=1,2,\cdots,N;\quad k=1,2,\cdots,K$$

**Maximization step**

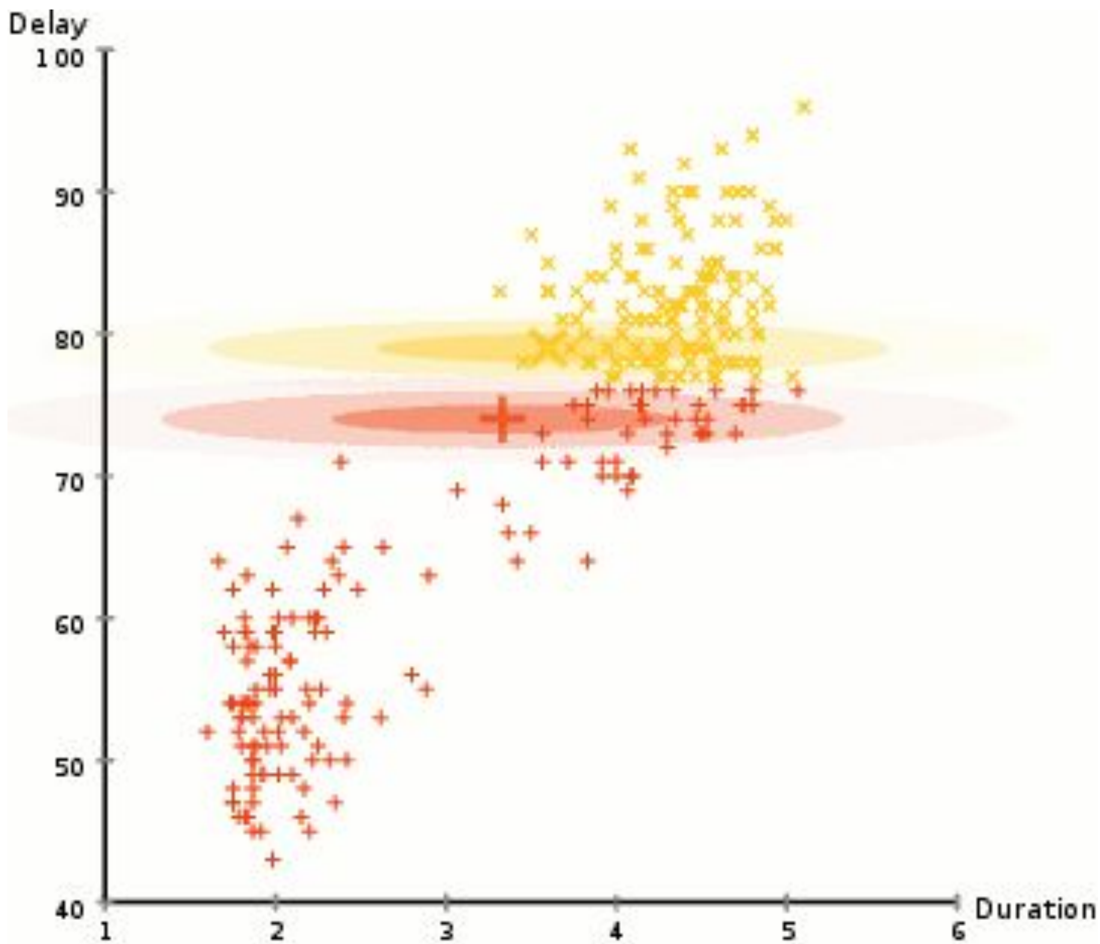
$$\hat{\mu}_k = \frac{\sum_{j=1}^{N}\hat{\gamma}_{jk}y_j}{\sum_{j=1}^{N}\hat{\gamma}_{jk}}, \quad k=1,2,\cdots,K$$

$$\hat{\sigma}_k^2 = \frac{\sum_{j=1}^{N}\hat{\gamma}_{jk}(y_j-\mu_k)^2}{\sum_{j=1}^{N}\hat{\gamma}_{jk}}, \quad k=1,2,\cdots,K$$

$$\hat{\alpha}_k = \frac{n_k}{N} = \frac{\sum_{j=1}^{N}\hat{\gamma}_{jk}}{N}, \quad k=1,2,\cdots,K$$

- 3. **Learning Problem**(Training Problem):

**EM Algorithm**

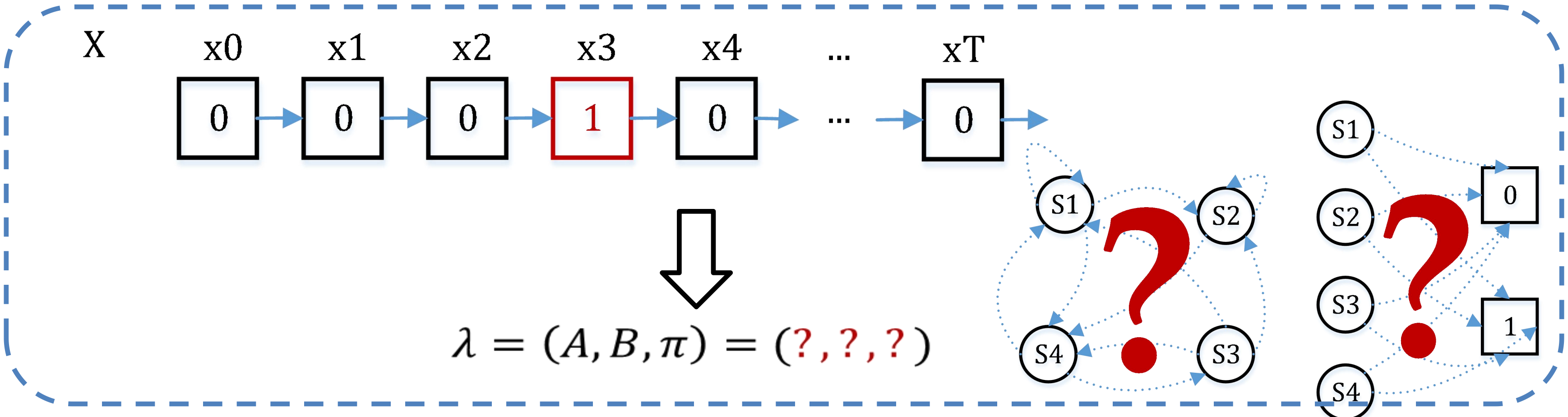



GMM收敛动图(from wikipedia)

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

Given a set of observation sequences $X = (x1, x2, \ldots xT)$, How do we adjust the model parameters to maximize $P(X \mid \lambda)$?




$$\lambda = (A, B, \pi) = (?, ?, ?)$$

$$\hat{\lambda} = \arg\max_{\lambda} logP(X \mid \lambda)$$
$$= \arg\max_{\lambda} logP(X \mid Z, \lambda)P(Z \mid \lambda)$$

体会这个过程：

一组参数$\lambda = (A, B, \pi)$先产生一个分布$P(Z \mid \lambda)$，代表邵老师拖延症的状态(非概率刻画)，这种拖延症状态又决定了邵老师是否去学车$logP(X \mid Z, \lambda)$，（去或不去，离散）

完全类似于GMM中过程：用参数$\theta = (\alpha_k, \mu_k, \sigma_k)$先得到所有点属于各个高斯函数的概率分布（概率刻画），再通过这个分布计算各点属于这个GMM的概率(连续)。

# 2.2 Three Classic Problems

- 3. **Learning Problem**(Training Problem):

**Baum-Welch Algorithm（EM思想）**

2．EM 算法的 E 步：求 $Q$ 函数 $Q(\lambda,\bar{\lambda})$[①]

$$Q(\lambda,\bar{\lambda})=\sum_I \log P(O,I\mid\lambda)P(O,I\mid\bar{\lambda}) \qquad (10.33)$$

其中，$\bar{\lambda}$ 是隐马尔可夫模型参数的当前估计值，$\lambda$ 是要极大化的隐马尔可夫模型参数.

$$P(O,I\mid\lambda)=\pi_{i_1}b_{i_1}(o_1)a_{i_1i_2}b_{i_2}(o_2)\cdots a_{i_{T-1}i_T}b_{i_T}(o_T)$$

于是函数 $Q(\lambda,\bar{\lambda})$ 可以写成：

$$Q(\lambda,\bar{\lambda})=\sum_I \log\pi_{i_1}P(O,I\mid\bar{\lambda}) + \sum_I\left(\sum_{t=1}^{T-1}\log a_{i_ti_{t+1}}\right)P(O,I\mid\bar{\lambda})+\sum_I\left(\sum_{t=1}^{T}\log b_{i_t}(o_t)\right)P(O,I\mid\bar{\lambda}) \qquad (10.34)$$

式中求和都是对所有训练数据的序列总长度 $T$ 进行的.

$$a_{ij}=\frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)}$$

$$b_j(k)=\frac{\sum_{t=1,o_t=v_k}^{T}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)}$$

$$\pi_i=\gamma_1(i)$$

- 这是省去大量推导过程的结果。形式非常简单。
- 注意右边$\gamma_t(j),\xi_t(i,j)$是第一个问题的计算结果

# Framework

**1. Markov Chains and Markov Property**

- Examples of Markov Chains
- Something about Markov Property

**2. Hidden Markov Models**

- Definition and Examples
- Three classic Problems
  - A. Evaluation Problem: Forward/Backward Algorithm
  - B. Decoding Problem: Viterbi Algorithm (Dynamic Programming)
  - C. Learning Problem: Baum-Welch Algorithm(EM Algorithm & GMM)

**3. Applications of HMM**

- Speech Recognition
- On-Line Hand Written Digits
- Computational Biology

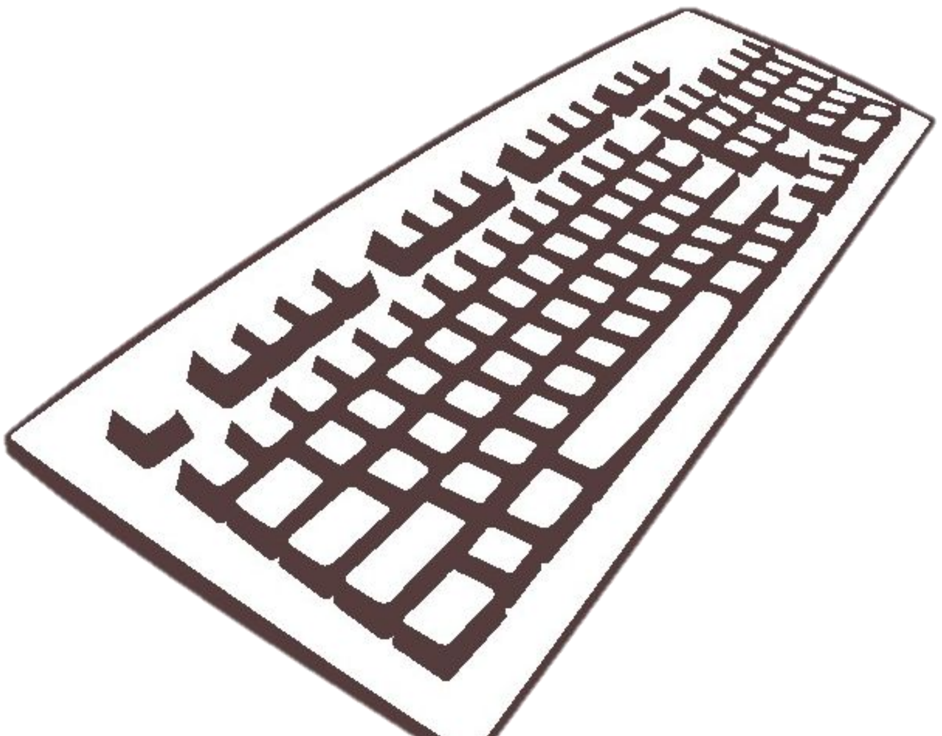
**4. Other Issues of HMMs**

- Types of HMMs
- Implementation Issues

**5. Discussion: Generalize to Conditional Random Field**

# 3.1 汉语拼音识别

*“shaolaoshiaishuijiao”*



# 3.1 汉语拼音识别

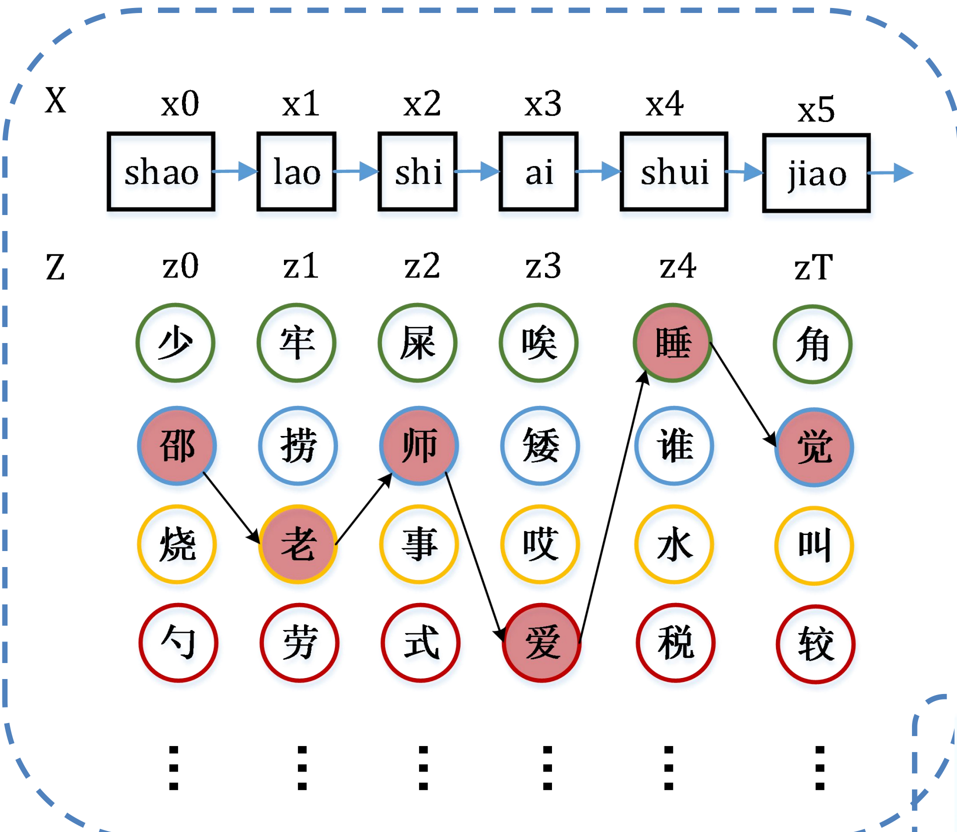
例子：爱睡觉的邵老师


没错！我们的男神邵老师又来了，他英俊潇洒风流倜傥高大威猛，而又不失斯文儒雅大方得体温柔体贴。

邵老师很少睡觉。因为他不睡则已，一睡惊人！

# 3.1 汉语拼音识别

1. 输入的汉语拼音系列：*"shaolaoshiaishuijiao"*
2. 基于规则截断，成为"shao lao shi ai shui jiao"。
3. 作为HMM的观测序列$X = (x1, x2, \ldots xT)$输入，要求其隐藏序列$Z = (z1, z2, \ldots zT)$，每个$zi$代表一个汉字。模型$\lambda = (A, B, \pi)$由大量统计得到。问题转换为Decoding Problem




Viterbi algorithm

利用动态规划，递推求解，得到概率最大的$Z$序列。

*Bingo !*

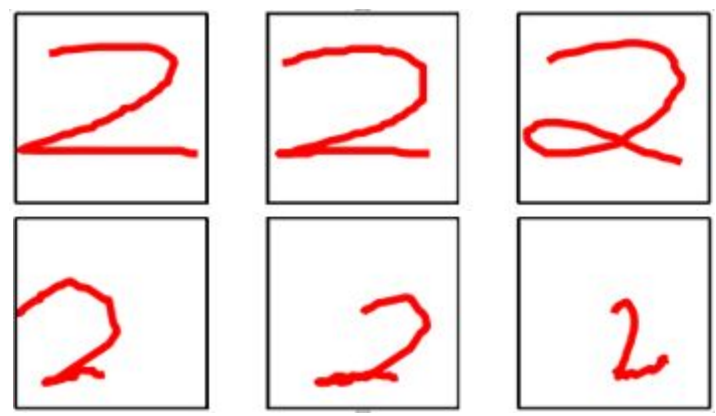
# 3.2 On-line Hand Written Digits Generation

描述：

用45个手写的数字“2”，将每个数字2细分为一系列线段的连接。每个线段有16个角度，对应观测变量$X$的状态空间。定义隐藏变量$Z$有16个。隐藏变量不能解释。

1. Training: 用EM的思想，用Baum-Welch算法求得参数$\lambda = (A, B, \pi)$

2. Generation: 用模型$\lambda = (A, B, \pi)$，求得概率比较大的几种生成序列：$Z = (z1, z2, \dots zT)$

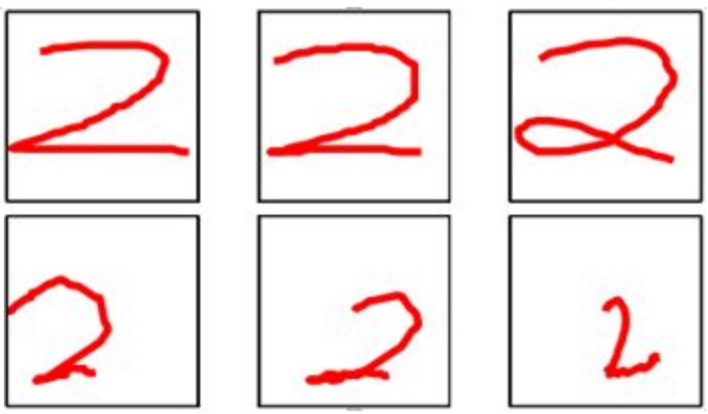


第一行是训练数据：手写体
第二行是模型自动生成的数字

**On-line Hand Written Digits**

# 3.2 On-line Hand Written Digits Generation

讨论：从在线手写体中看HMM特性



第一行是训练数据：手写体
第二行是模型自动生成的数字

**On-line Hand Written Digits**

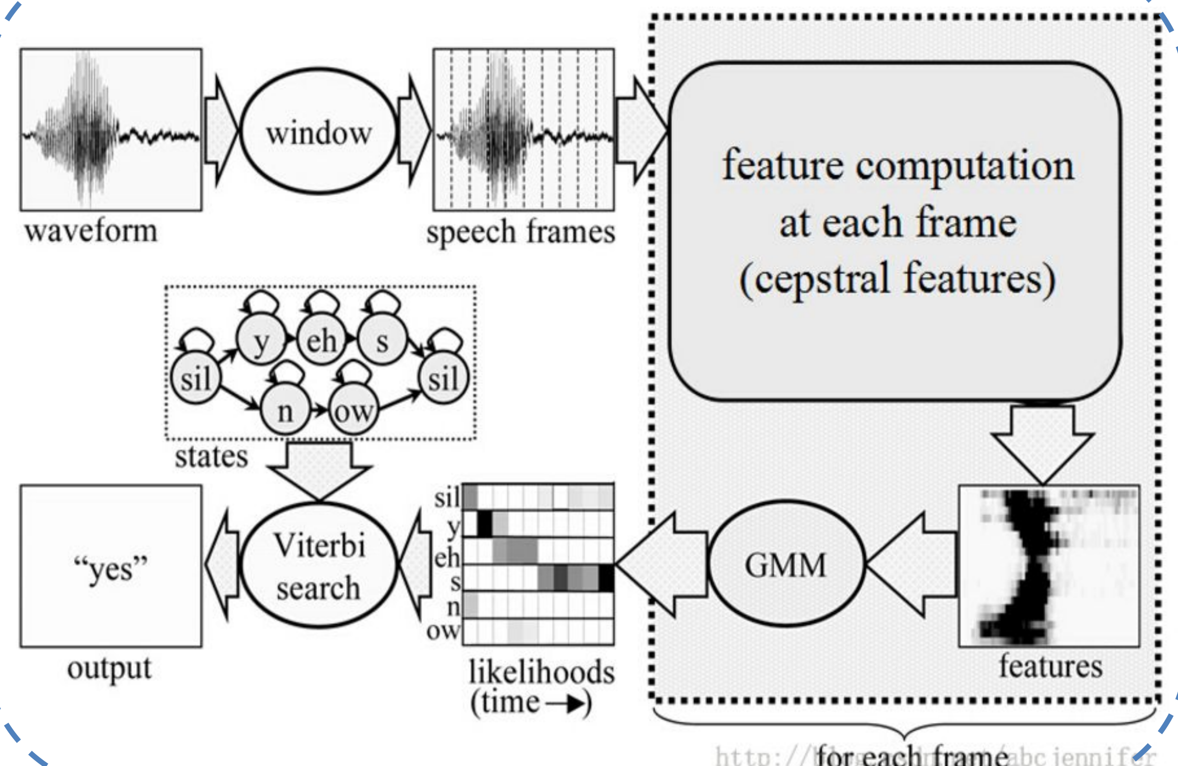
HMM相比其他模型最大的特性是：
ability to exhibit some degree of invariance to local warping (compression and stretching) of the time axis.

数字‘2’由两个组成部分组成，即上面的环状和下面的一横。它们可能随着不同人的习惯而使得其比例、出现早晚有所区别。HMM能接受这种区别。

# 3.3 语音识别

描述：

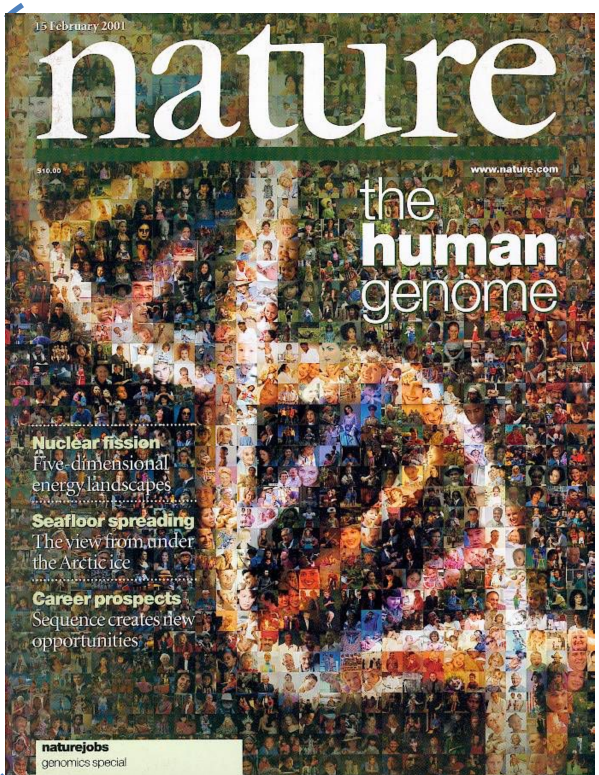
和汉语拼音的例子类似。语音frames提取MFCCs特征，用这些特征做GMM得到最可能的观测状态序列$X = (x1, x2, \ldots xT)$，预测概率最大的文字序列$Z = (z1, z2, \ldots zT)$.

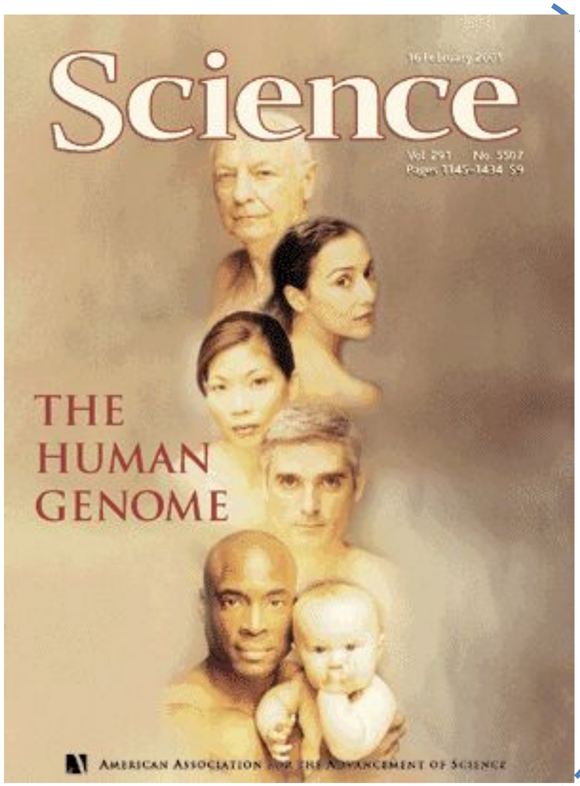



MFCCs（Mel Frequency Cepstral Coefficents）是一种在自动语音和说话人识别中广泛使用的特征。它是在1980年由Davis和Mermelstein提出的。从那时起。在语音识别领域，MFCCs在人工特征方面可谓是鹤立鸡群，一枝独秀，从未被超越（至于说Deep Learning的特征学习那是后话了）。

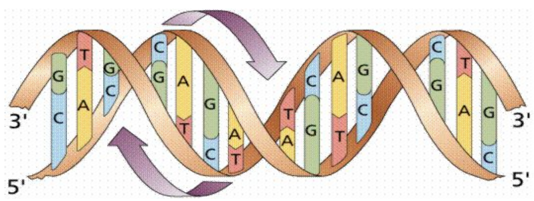
# 3.4 Computational Biology

This is one of the most challenging and interesting problems in computational biology at the moment. With so many genomes being sequenced so rapidly, it remains important to begin by identifying genes computationally.

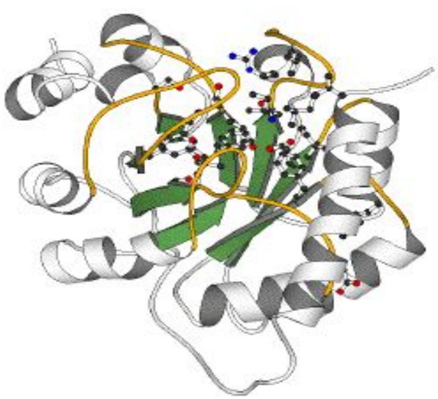


# 3.4 Computational Biology




DNA → transcription → mRNA → translation → Protein



CCTGAGCCAACTATTGATGAA

↓

CCUGAGCCAACUAUUGAUGAA

↓

PEPTIDE

# Framework

**1. Markov Chains and Markov Property**

- Examples of Markov Chains
- Something about Markov Property

**2. Hidden Markov Models**

- Definition and Examples
- Three classic Problems
  - A. Evaluation Problem: Forward/Backward Algorithm
  - B. Decoding Problem: Viterbi Algorithm (Dynamic Programming)
  - C. Learning Problem: Baum-Welch Algorithm(EM Algorithm & GMM)

**3. Applications of HMM**

- Speech Recognition
- On-Line Hand Written Digits
- Computational Biology

**4. Other Issues of HMMs**

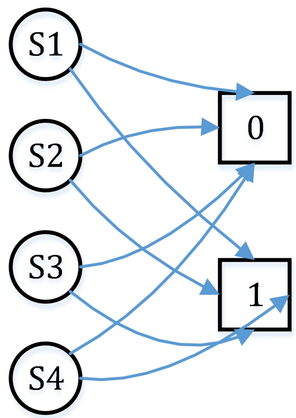
- Types of HMMs
- Implementation Issues

**5. Discussion: Generalize to Conditional Random Field**

# 4.1 Other Issues

- Types of HMMs

我们一直讨论的离散情况：

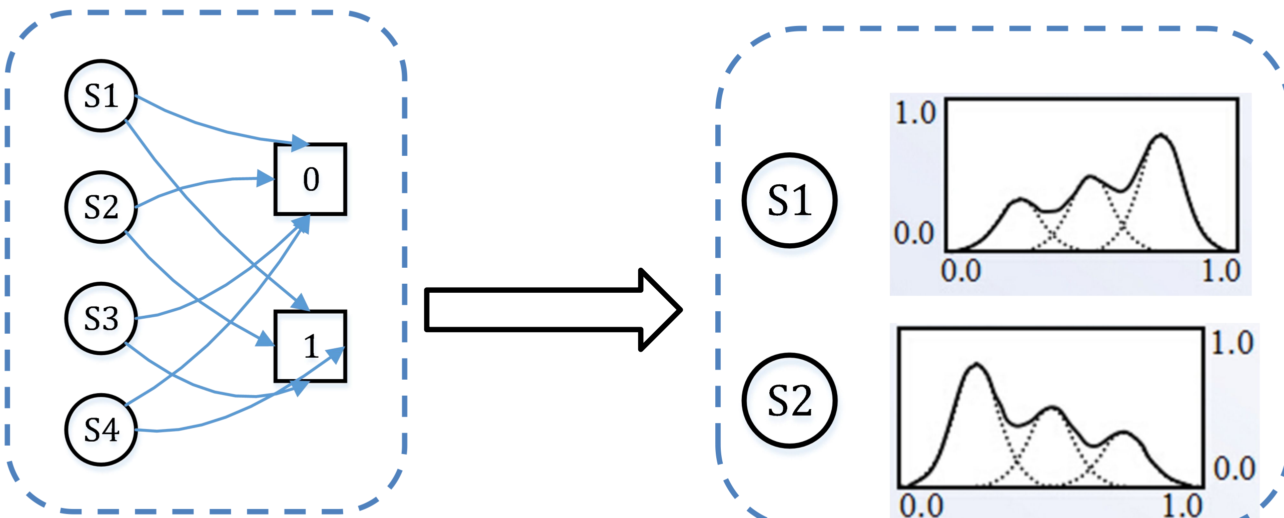



Emission diagram

$$B = \begin{pmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{pmatrix}$$

**Emission matrix**

0和1代表邵老师是否去开车。若换一个问题，换为“邵老师每天回家的时间。”这就不是一个离散的概率表能刻画了，而是连续的分布了。



- Types of HMMs

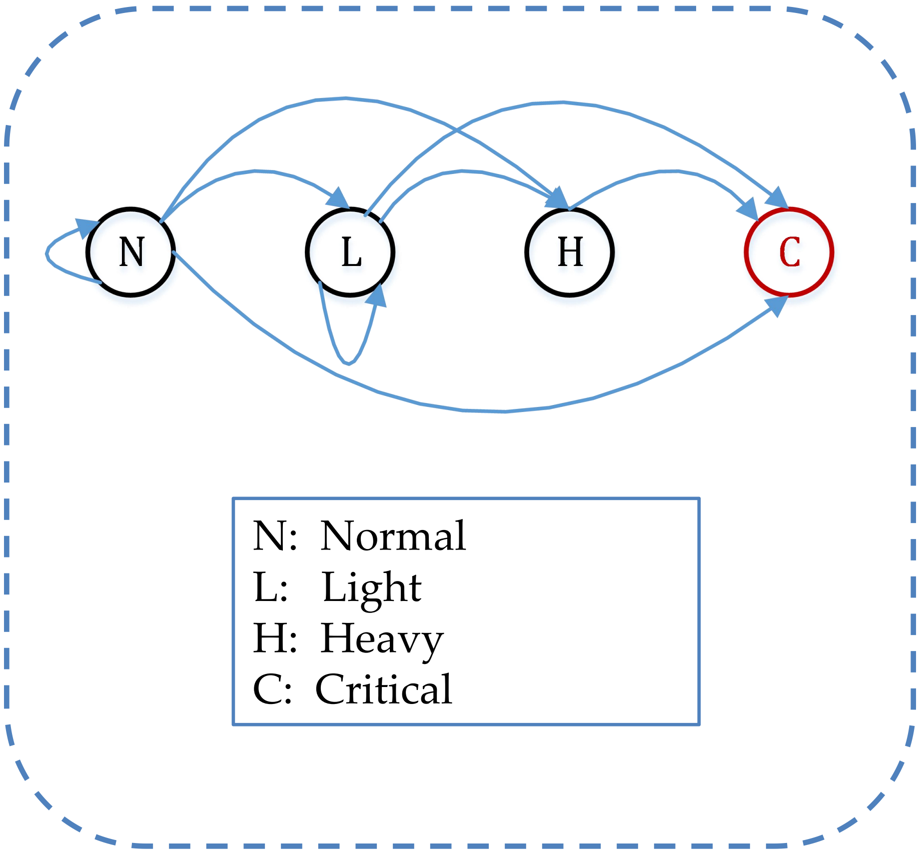
更特殊一点，当观测序列与隐藏序列都成为连续分布后，若这个分布为高斯分布，则HMM转化为LDS（Linear Dynamical Systems）,具体可看《PRML》第十三章。

这种情况下，之前概率计算式中的求和符号将变为积分符号，要引入新的数学方法求解。可看pluskid大神的博客*Free Mind*上面"漫谈 HMM：Kalman/Particle Filtering"

# 4.1 Other Issues

➢ Types of HMMs

Left to right Hidden Markov model




这就是一个有限状态机

在这种情况下邵老师在拖延症作用下，心理负担只会越来越重，不会逆转减轻

# 4.1 Other Issues

➢ Types of HMMs

由于时间关系，不能再讲马尔科夫的计算机Implementation部分。

简单提一下，由于概率迭代的原因，概率将越来越小，小到计算机精度表示不出来的范围，这种情况下需要乘上一个factor.

详情见tutorial上的scaling一节。

# Framework

**1. Markov Chains and Markov Property**

- Examples of Markov Chains
- Something about Markov Property

**2. Hidden Markov Models**

- Definition and Examples
- Three classic Problems
  - A. Evaluation Problem: Forward/Backward Algorithm
  - B. Decoding Problem: Viterbi Algorithm (Dynamic Programming)
  - C. Learning Problem: Baum-Welch Algorithm(EM Algorithm & GMM)

**3. Applications of HMM**

- Speech Recognition
- On-Line Hand Written Digits
- Computational Biology

**4. Other Issues of HMMs**

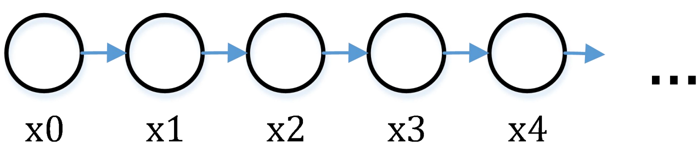
- Types of HMMs
- Implementation Issues

**5. Discussion: Generalize to Conditional Random Field**

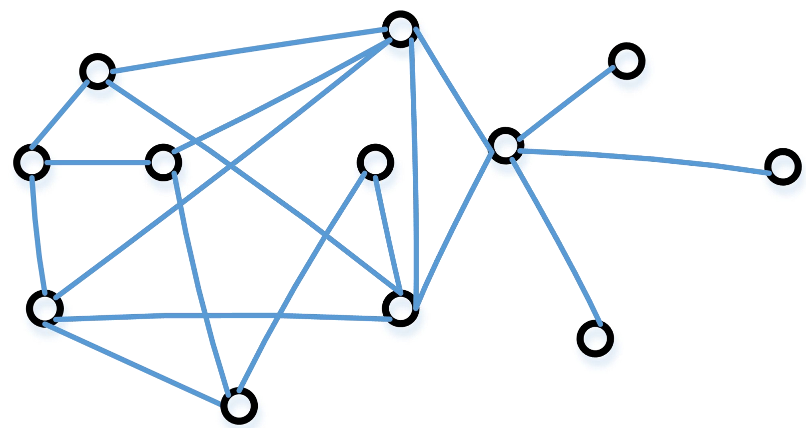
# 5. Discussion: Generalize to Conditional Random Field

在我看来：

条件随机场就是HMM的拓展，首先忽略马尔科夫性，即无后效性，然后将HMM中事件的概率描述用特征函数来代替。（在大多数问题中，这个特征函数还是以概率方式给出）




马尔科夫序列



条件随机场

*Thanks*

Chongming Gao
Yingcai Experimental School
gchongming@126.com