Modèles de Markov Cachés

Hidden Markov Model
(HMM)

Introduction aux chaînes de Markov:

Généralement, un processus stochastique est une suite d'expériences dont le résultat dépend du hasard.

En certains temps 0, 1, 2, ..., t, observons un système. Celui-ci peut se trouver dans l'un des états d'une collection finie d'états possibles.

L'observation du système est ainsi considérée comme une expérience dont le résultat (aléatoire) est l'état dans lequel se trouve le système.

Un **processus stochastique** est un phénomène temporel où intervient le hasard, *i.e.* une variable aléatoire X(t) évoluant en fonction du temps.

Ex: Une suite de lancers de dés 1,3,2,5,3,6,2,4

Un processus stochastique est dit **markovien** si son évolution ne dépend pas de son passé, mais uniquement de son état présent.

Si à l 'instant ti, on a observé la réalisation O_i de X(t) alors:

$$P(X(t_n) \le o_n \mid X(t_{n-1}) = o_{n-1}, \ldots, X(t_1) = o_1) = P(X(t_n) \le o_n \mid X(t_{n-1}) = o_{n-1})$$

Une chaîne de Markov est un processus aléatoire portant sur un nombre fini d'états, avec des probabilités de transition **sans mémoire**.

L 'état courant d 'un système contient toute l 'information pour prédire son état futur.

Exemple typique :

Tirage avec remise différée: Dans une urne, on a placé 2 boules rouges (notées R) et 2 boules noires (notées N). On effectue des tirages successifs, avec remise différée d'un tour:

- au 1er tirage : on enlève une boule choisie au hasard dans l'urne
- au i-eme tirage : on enlève une boule choisie au hasard parmi les boules de l'urne et on remet la boule tirée au (i-1) ème tirage.

La suite de variables aléatoires considérées est Xn = la couleur de la boule prise au n ème tirage.

Si on connait la couleur de la boule tirée à la n-ième étape, alors on connait la constitution de l'urne lorsqu'on effectuera le n+1-ième tirage et donc on a tout ce qui faut pour prédire de la façon la plus précise possible ce qui se passera la n+1-ième tirage :

- sachant que la n-ème boule tire est rouge, la probabilité de tirer une boule rouge au n+1 ème tirage est de 1/3;
- sachant que la n-ème boule tire est noire, la probabilité de tirer une boule rouge au n+1 ème tirage est de 2/3.

Le fait de connaître, en plus de la couleur de la n-ième boule tirée, le résultat des informations sur les n-1 premiers tirages ne permet pas de modifier les prédictions précédentes.

Dans ce qui suit, on parle de **chaîne de Markov** à processus X(t) est <u>discret</u>.

$$P(X(t_n)=o_n \mid X(t_{n-1})=o_{n-1},\ldots,X(t_1)=o_1) = P(X(t_n)=o_n \mid X(t_{n-1})=o_{n-1})$$

qui devient par abus de notation:

$$P(X(t_n)=o_n \mid X(t_{n-1})=o_{n-1}) = P(o_n \mid o_{n-1})$$

Propriétés d ’une chaîne de Markov

Probabilité d ’une séquence $O=(o_1, ...,o_n)$

$$P(o_n,o_{n-1},...,o_1) = P(o_n|o_{n-1},...,o_1) \times P(o_{n-1},...,o_1)$$

$$P(o_n,o_{n-1},...,o_1) = P(o_n|o_{n-1}) \times P(o_{n-1}|o_{n-2},...,o_1) \times P(o_{n-2},...,o_1)$$

.....

$$P(o_n,o_{n-1},...,o_1) = P(o_n|o_{n-1}) \times P(o_{n-1}|o_{n-2}) \times ... \times P(o_2|o_1) \times P(o_1)$$

$$P(O) = P(o_n,o_{n-1},...,o_1) = P(o_1) \times \prod_{t=2}^{n} P(o_t|o_{t-1})$$

Représentation d ’une chaîne de Markov

Formellement, il suffit de définir:

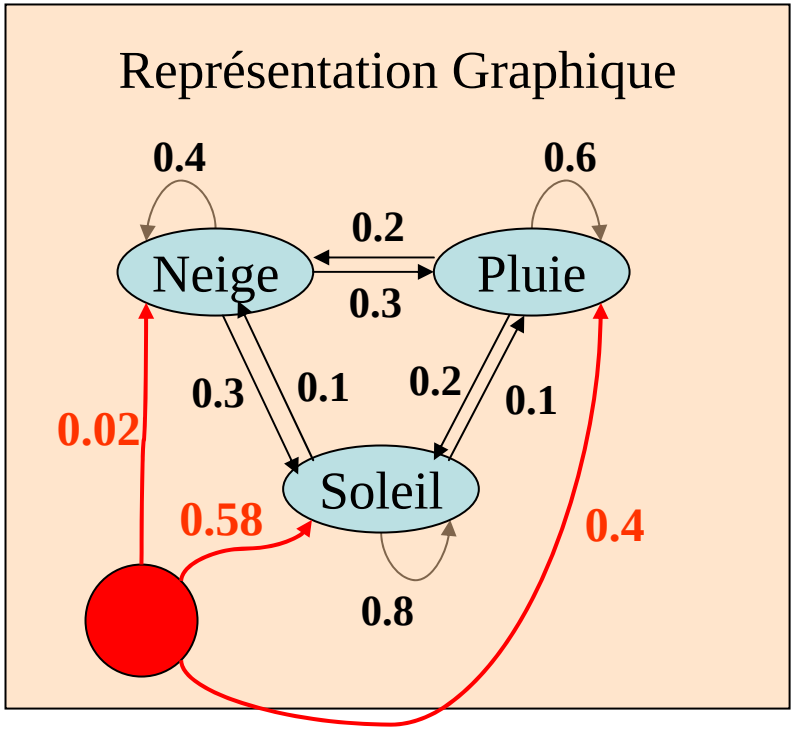
- L’espace d’états: l’alphabet $\Sigma=\{o_1,...,o_m\}$ des réalisations possibles de X(t)
- La matrice de transition $A=\{a_{i,j}= P(o_j|o_i)\}$
- Les probabilités de départ $\Pi=\{\pi_i=P(o_i)\}$

Exemple: X(t):Temps

$\Sigma=\{$‘ neige ’,‘ pluie ’, ’soleil ’$\}$

$$A=\begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{pmatrix}$$

$\Pi = \{0.02 , 0.4, 0.58\}$



$$\sum_{j=1}^{m} a_{ij} = 1 \qquad \sum_{i=1}^{m} \pi_i = 1$$

Exemple d ’application d ’une chaîne de Markov

Application:

Quelle est la probabilité qu’il fasse ‘ Soleil ’, 5 jours de suite ?

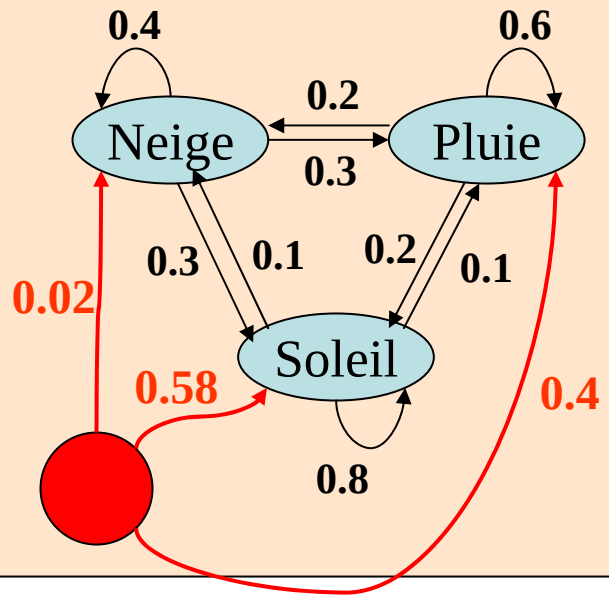
P(Soleil, Soleil, Soleil, Soleil, Soleil)=

*0.58*0.8*0.8*0.8*0.8 =0.2375*

P(Neige, Neige, Pluie, Pluie, Soleil)=

*0.02*0.4*0.3*0.6*0.2=0,000288*

Représentation Graphique



$$\sum_{j=1}^{m} a_{ij} = 1 \qquad \sum_{i=1}^{m} \pi_i = 1$$

Prévision à l ’aide d’une chaîne de Markov :

Système d’événements complet :

$$P(X(t)=o_i)=\sum_{j=1}^{m}\left[P(X(t)=o_i \mid X(t-1)=o_j)\times P(X(t-1)=o_j)\right]$$

Hypothèse de stationnarité :

$$P(X(t)=o_i \mid X(t-1)=o_j)=P(o_i \mid o_j)=a_{j,i}$$

Quelle est la probabilité qu ’il fasse ‘ Soleil ’ dans 3 jours ?

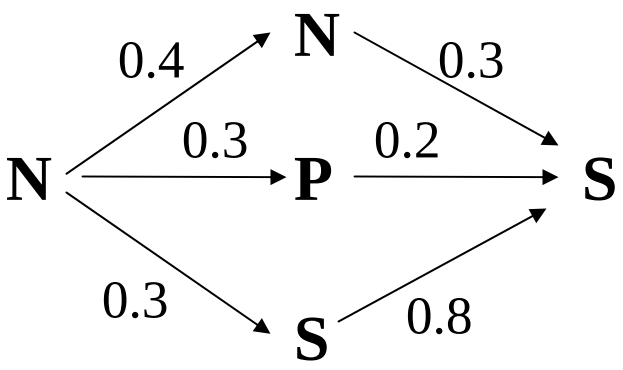
$$P(X(3)='S')=P('S' \mid 'N')\times P(X(2)='N')+P('S' \mid 'P')\times P(X(2)='P')+\cdots$$
$$P('S' \mid 'S')\times P(X(2)='S')$$

…

$$P(X(3)='S')=0.3*(0.4*0.02+0.2*0.4+0.1*0.58)+0.2*(0.3*0.02+\ldots$$
$$0.6*0.4+0.1*0.58)+0.8*(0.3*0.02+0.2*0.4+0.8*0.58)=0.5446$$

$$P(X(t_3)=o_k \mid X(t_1)=o_i)=\sum_{j=1}^{m} P(X(t_3)=o_k \mid X(t_2)=o_j)P(X(t_2)=o_j \mid X(t_1)=o_i)$$

Quelle est la probabilité qu'il fasse ' Soleil' dans 3 jours sachant qu 'il neige aujourd 'hui ?



$$P(X(3)='S' \mid X(1)='N')=P('S' \mid 'N')\times P('N' \mid 'N')+P('S' \mid 'P')\times P('P' \mid 'N')+\cdots$$

$$P('S' \mid 'S')\times P('S' \mid 'N')=0.3*0.4+0.2*0.3+0.8*0.3=0.42$$

Modèles de Markov Cachés

Les modèles de Markov cachés sont utilisés pour modéliser des séquences d'observations. Ces observations peuvent être de nature discrète (par exemples les caractères d'un alphabet fini) ou continue (fréquence d'un signal, température).

Une chaîne de Markov cachée est une chaîne de Markov dont les états ne sont pas déterminés mais qui génèrent une suite de variables aléatoires (suite d'observations) indépendantes deux à deux.

- Un modèle de Markov caché est un modèle stochastique particulier, il est composé de deux suites de variables aléatoires, la première est cachée et la deuxième est observable.

- La suite **cachée** correspond à la suite d'états q_1, q_2…q_r où les q_i prennent leurs valeurs parmi l'ensemble des N états (i indice temps) du modèle.

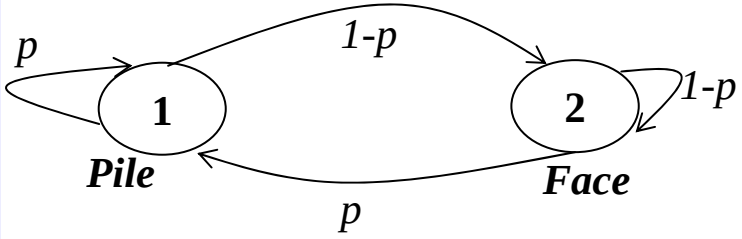
- La suite **observable** correspond à la suite d'observations O_1, O_2 …O_T où les O_i sont aussi en fonction du temps et se réalisent parmi un ensemble de M symboles.

Exemple introductif :

On peut illustrer la différence entre un modèle de Markov et un modèle de Markov caché à l'aide de l'exemple classique de lancée de pièce. La suite d'observation de ce problème est formée de « pile » et « face ». supposons que derrière un rideau il y a une autre personne qui manipule une pièce de monnaie, elle ne vous tient pas au courant de ce qu'elle fait réellement, elle vous donne seulement les résultats : pile ou face.

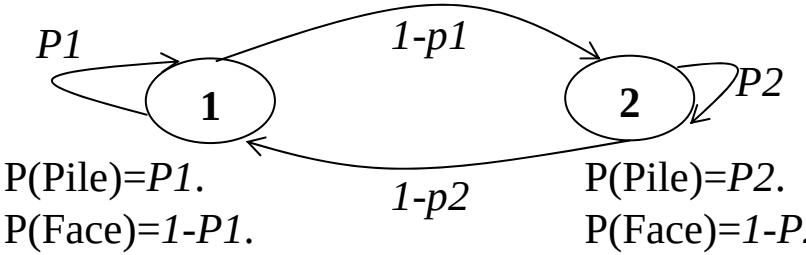
Le problème est comment construire un HMM pour expliquer la suite d'observations pile et face. Le modèle de Markov se compose de deux états (Fig1). L'état 1 correspond à « pile » et l'état 2 à « « face ».

Si la probabilité d'observer « pile » à l'état 1 est *p*, alors la probabilité d'observer « face » à l'état 2 est *1-p.*



O : PPFFPFPPFFP…
Q : 11221211221…

Fig 1 : chaîne de Markov observable



O : PPFFPFPPFFP…
Q : 21122212212…

Fig2 : *chaîne de Markov cachée à deux états*

Par contre dans le modèles de Markov cachés *Fig2*, on peut observer dans chaque état, l'événement « pile » et « face », il deviens impossible de déduire la suite d'états Q à partir de la suite d'observations O, c'est ce qui procure le caractère caché du modèle.

Exemple 2 :

- L'exemple des urnes (Fig3) reflète parfaitement les deux composantes du processus stochastiques d'un HMM construit comme suit :

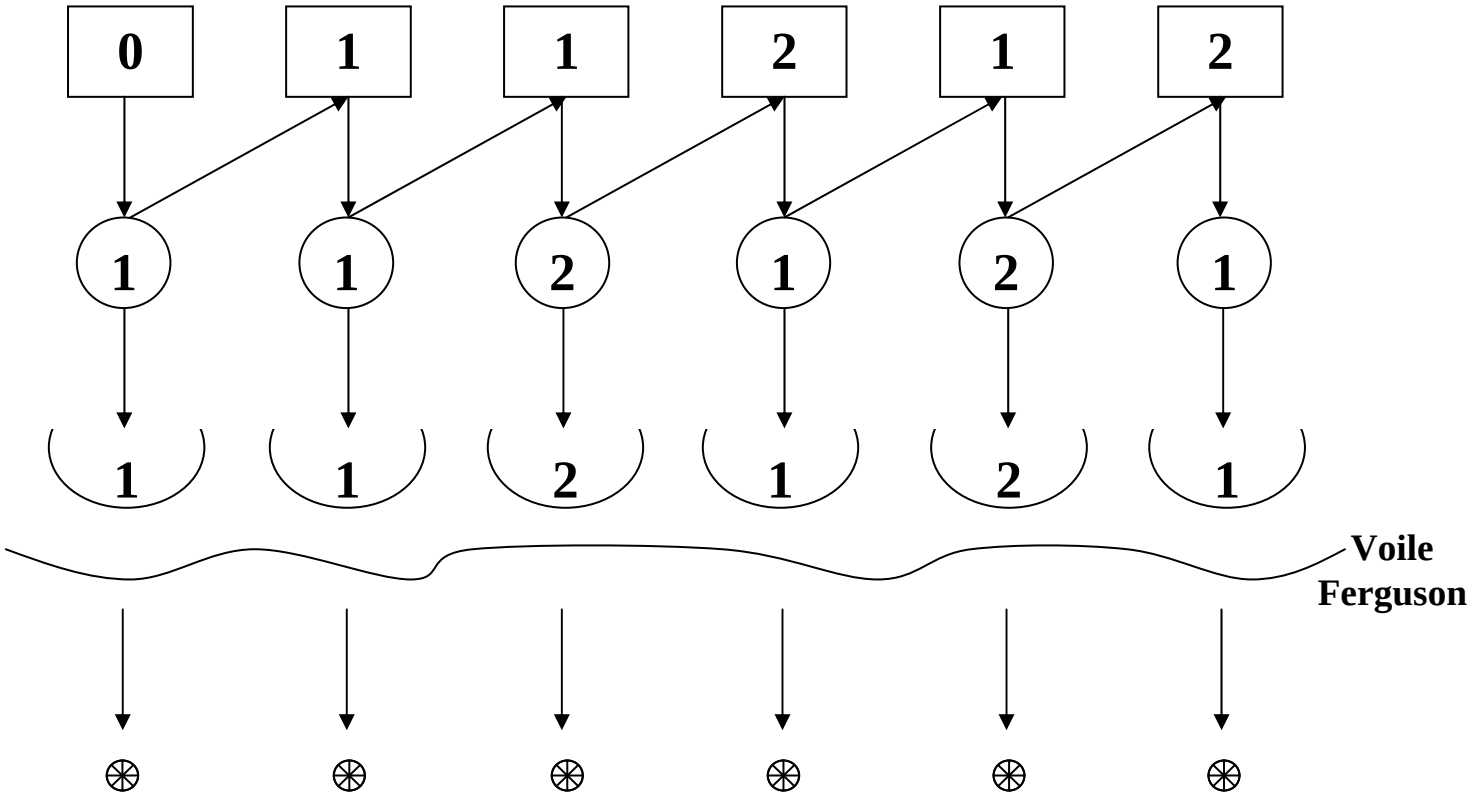
- Deux urnes : urne1 et urne2. Chaque urne a son propre mélange de balles rouges et noires ($b_1(r)$, $b_1(n)$, $b_2(r)$, $b_2(n)$) : les fractions des balles rouges et noires respectivement dans l'urne 1 et l'urne 2.

- Trois gobelet : $gobelet_0$, $gobelet_1$, $gobelet_2$ contenants des pierres portant des marques (a_{01}, a_{02}, a_{11}, a_{12}, a_{21}, a_{22} : les fractions des pierres marquées « état1, état2 » respectivement dans $gobelet_0$, $gobelet_1$, $gobelet_2$).

- Générons uns suite d'observations de couleurs notée $O=(O_1, \ldots O_r)$ comme suit : tirons aléatoirement une pierre du gobelet0 ; sa marque est appelé état s_1. Tirons une balle aléatoirement de l'urne s_1 ; sa couleur est O_1.

- Maintenons tirons une pierre du gobelet s_1 ; sa marque est appelé état s_2.
- Continuons dans cette voie, en utilisant l'état courant pour obtenir à la fois l'observation courante et l'état suivant jusqu'à un total de T observations.
- Cette suite d'observations est observable et donne une information probabiliste de la suite des pierres $S = (s_1, s_2, \ldots, s_T)$ qui est cachée par le voile de Ferguson qui empêche l'observateur de voir les gobelets.

Modèle de Markov caché du 1er ordre à deux états.

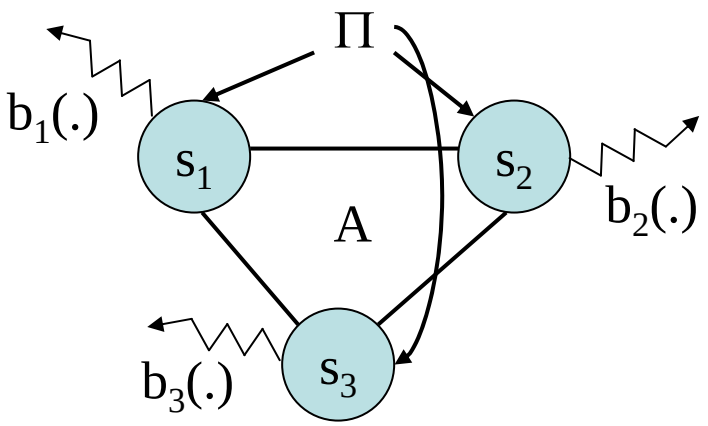


Remarque : un nombre fini N des états est possible, dans ce cas, il faut N urnes et N+1 gobelets.

Définition d ’un modèle de Markov caché (HMM)

Formellement, un HMM est défini par: $H=<\Sigma,S,\Pi,A,B>$

- L’alphabet $S=\{s_1,...,s_N\}$ des états de la chaîne de Markov, q_t états
- La matrice de transition $A=\{a_{i,j}= P(s_j|s_i)\}$
- Les probabilités de départ $\Pi=\{\pi_i=P(s_i)\}$: prob être initialement dans un état.
- L’alphabet $\Sigma=\{v_1,...,v_M\}$ des symboles émis par les s_i pour un HMM discret
- Les probabilités d ’émission $B=\{b_i(v_k)=P(v_k|s_i)\}$, distribution symb obs dans un état.



- N nombre d'états du modèle, États : $S = \{s_1, s_2, ..., s_N\}$
- État au temps t : $q_t \in S$
- M nombre symboles observations.
- Distribution de prob. de transition d'états:

$$a_{ij} = P(q_{t+1} = s_j \,|\, q_t = s_i) \quad 1 \le i, j \le N$$

- Distribution de prob. d'obs d'un symb à l'état j : $B = \{b_j(k)\}$

$$b_j(k) = P(x_k \,|\, q_t = s_j) \quad 1 \le j \le N, 1 \le k \le M$$

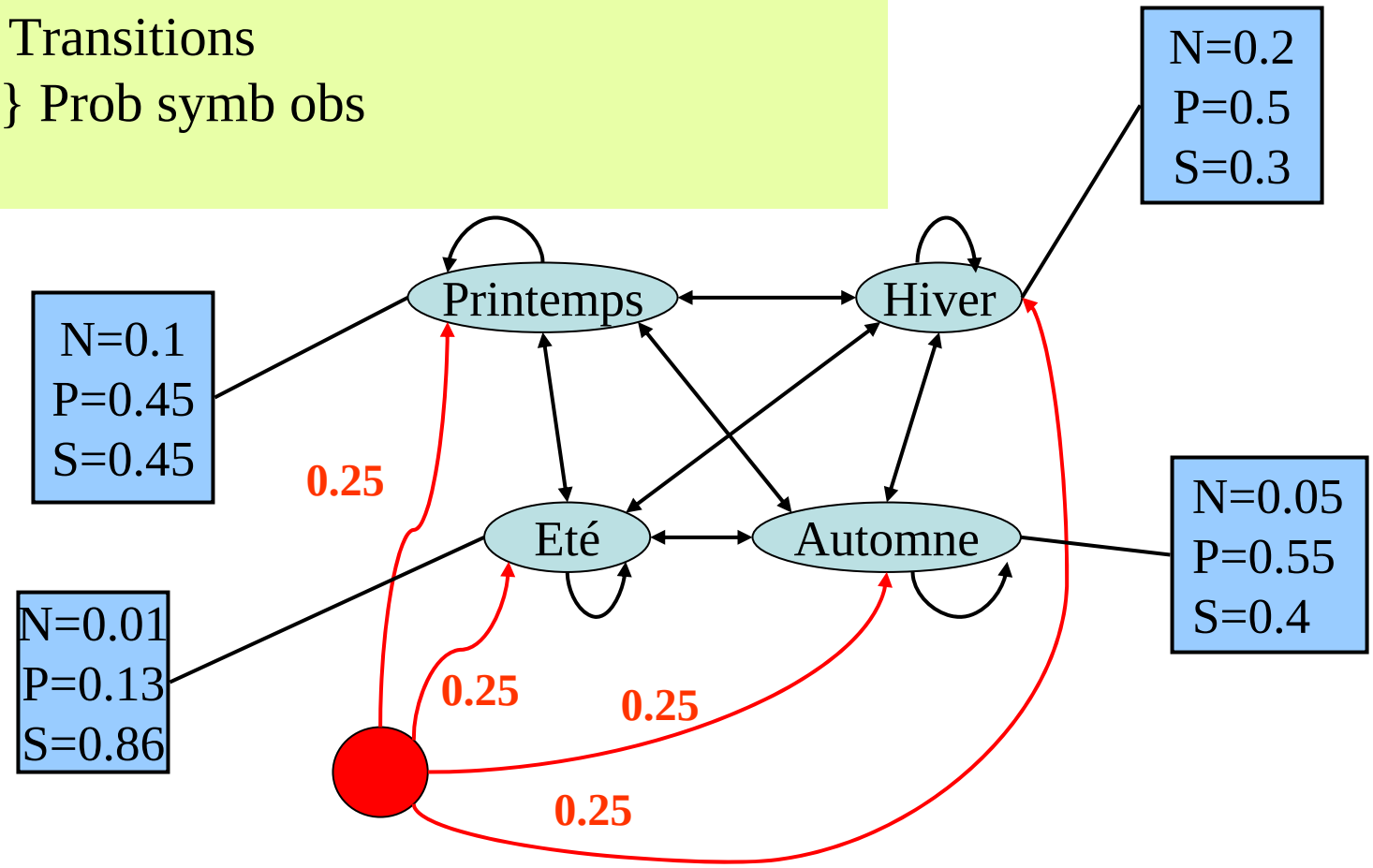
- Distribution des états initiaux : $\Pi = \{\pi_i\}$

$$\pi_i = P(q_1 = s_i), \quad 1 \le i \le N$$

Un HMM est décrit par : $\lambda = (\pi, A, B)$

Un exemple de HMM

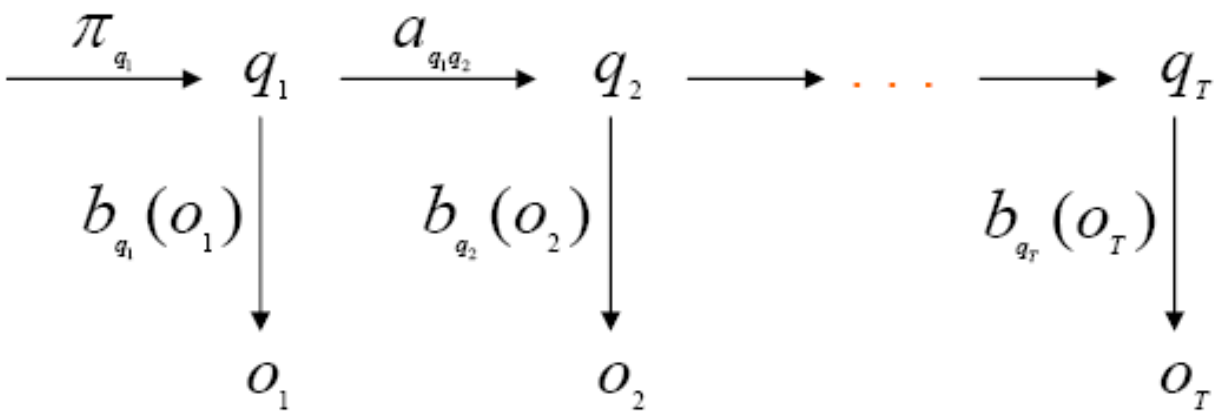
S={‘ Printemps ’, ’Eté ’, ‘ Automne ’, ’Hiver ’}
Σ={‘ N ’, ’P ’, ’S ’}
A={$a_{i,j}$} Transitions
B={$b_j(.)$} Prob symb obs



Génération des observations dans un HMM se fait comme suit:

1. Pas $t = 1$, Choix de l'état initial $q_1 = s_i$ avec la distribution π_i
2. choix de l'observation $O_t = V_k$ avec la distribution $b_i(k)$
3. transition au nouvel état, $q_{t+1} = s_j$ avec la probabilité a_{ij}
4. Pas $t = t + 1$. Si $t \prec T$ alors retour à l'étape 2, sinon fin de procédure.

Avec T la longueur d'une suite d'observations.



Problèmes de base des HMM

1. **Évaluation :**
   1. Problème: calculer la probabilité d'observation de la séquence d'observations étant donnée un HMM: $P(O|\lambda)$
   2. Solution: Forward Algorithm
2. **Décodage :**
   1. Problème: trouver la séquence d'états (estimation de la partie cachée) qui maximise la séquence d'observations
   2. Solution: Viterbi Algorithm
3. **Entraînement :**
   1. Problème: ajuster les paramètres du modèle HMM afin de maximiser la probabilité de générer une séquence d'observations à partir de données d'entraînement
   2. Solution: Forward-Backward Algorithm

Évaluation

- $P(O \mid \lambda)$ doit être évaluée pour toutes les séquences d'états possibles:

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

$$P(O, Q \mid \lambda) = P(O \mid Q, \lambda) P(Q \mid \lambda)$$

- Pour une séquence d'états donnée: $Q = q_1 q_2 \ldots q_T$

$$P(O \mid Q, \lambda) = b_{q_1}(o_1) b_{q_2}(o_2) \ldots b_{q_T}(o_T)$$

$$P(Q \mid \lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}$$

- Alors

$$P(O \mid \lambda) = \sum_{q_1 q_2 \cdots q_T} \pi_{q_1} a_{q_1 q_2} b_{q_1}(o_1) a_{q_2 q_3} b_{q_2}(o_2) a_{q_2 q_3} \ldots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

- Avec T observations et N états dans le modèle:
  - N^T possibles séquences d'états
  - Approximativement $2TN^T$ opérations requises
  - Pour T=100 et un HMM à 5 états $\approx 200 \times 5^{100}$ opérations

Algorithme Forward

- Une approche plus efficace pour évaluer $P(O \mid \lambda)$
- Définissons ….

$$\alpha_t(i) = P(o_1 o_2 \ldots o_t, q_t = i \mid \lambda)$$

- … comme étant la probabilité d'observations O_1 à O_t avec la séquence d'états qui se termine à l'état $q_t = s_i$ pour l'HMM λ

$$\alpha_t(i) = P(o_1 o_2 \ldots o_t, q_t = i \mid \lambda)$$

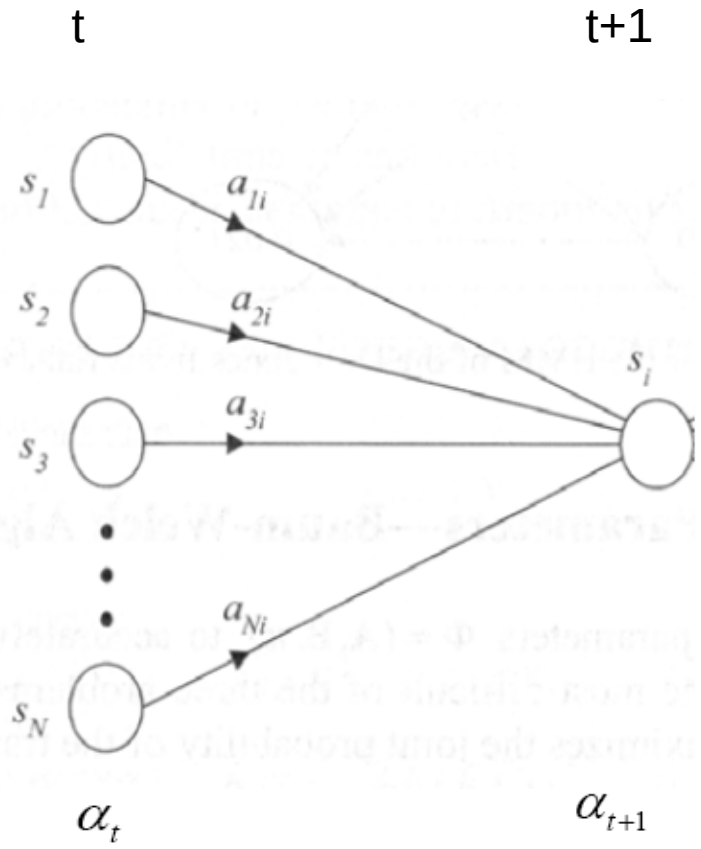
1. *Initialisation:* $\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \le i \le N$

2. *Induction:* $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), \quad 1 \le t \le T-1, \quad 1 \le j \le N$

3. *Terminaison:* $P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$

Avec T observations et N états, ceci requiert environ N^2T opérations

Algorithme Forward



$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), \quad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix}$$

Problèmes de base des HMM

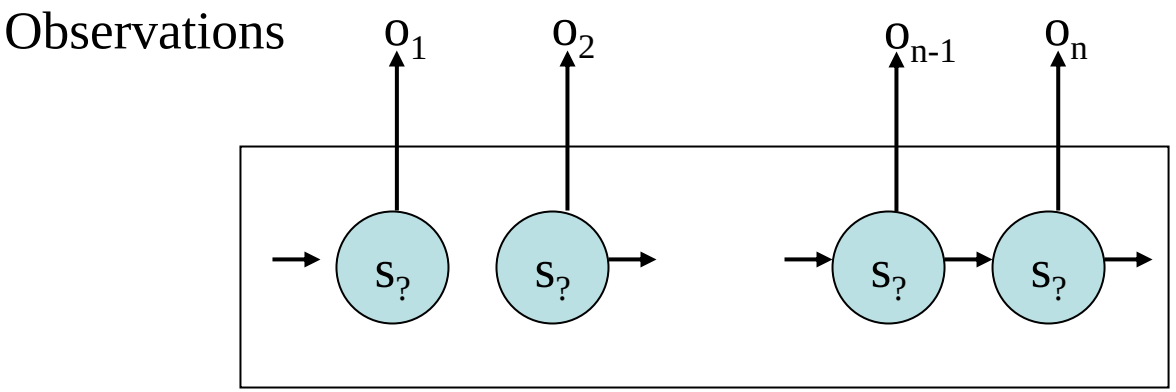
1. Évaluation:
   1. Problème: calculer la probabilité d'observation de la séquence d'observation étant donnée un HMM: $P(O \mid \lambda)$
   2. Solution: Forward Algorithm
2. Décodage:
   1. Problème: trouver la séquence d'état qui maximise la séquence d'observations
   2. Solution: Viterbi Algorithm
3. Entraînement:
   1. Problème: ajuster les paramètres du modèle HMM afin de maximiser la probabilité de générer une séquence d'observation à partir de données d'entraînement
   2. Solution: Forward-Backward Algorithm

Problème de décodage:

chercher les séquences d'états optimales (estimation de la partie cachée)

- Critère d'optimalité: choisir la séquence d'états qui maximise la probabilité $P(Q, O \mid \lambda)$ en utilisant l'algorithme de Viterbi

Algorithme de Viterbi



Problématique: Etant donné un HMM et une séquence observée O, quelle est la séquence d'états $s_{i(1)},\ldots,s_{i(n)}$ qui a la probabilité maximale d'avoir généré O ?

Problème de décodage:

- Définissons …

$$\delta_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P(q_1 q_2 \ldots q_{t-1}, q_t = s_i, o_1 o_2 \ldots o_t \mid \lambda)$$

- Comme étant la probabilité conjointe des observations O_1 à O_t et la séquence d'états $q_1 q_2 \cdots q_{t-1}$ se terminant à l'état $q_t = s_i$ étant donné l'HMM λ

- Nous pouvons montrer par induction que:

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(o_{t+1})$$

Algorithme de Viterbi

- Cumuler les probabilités des meilleurs chemins partiel $\delta_t(i)$ amenant à l'état s_i à l'instant *t* guidé par les t premières observations
- Garder trace des meilleures séquences : $\psi_t(i)$

Initialisation $\delta_1(i) = \pi_i b_i(o_1) \qquad \psi_1(i) = 0$

Induction

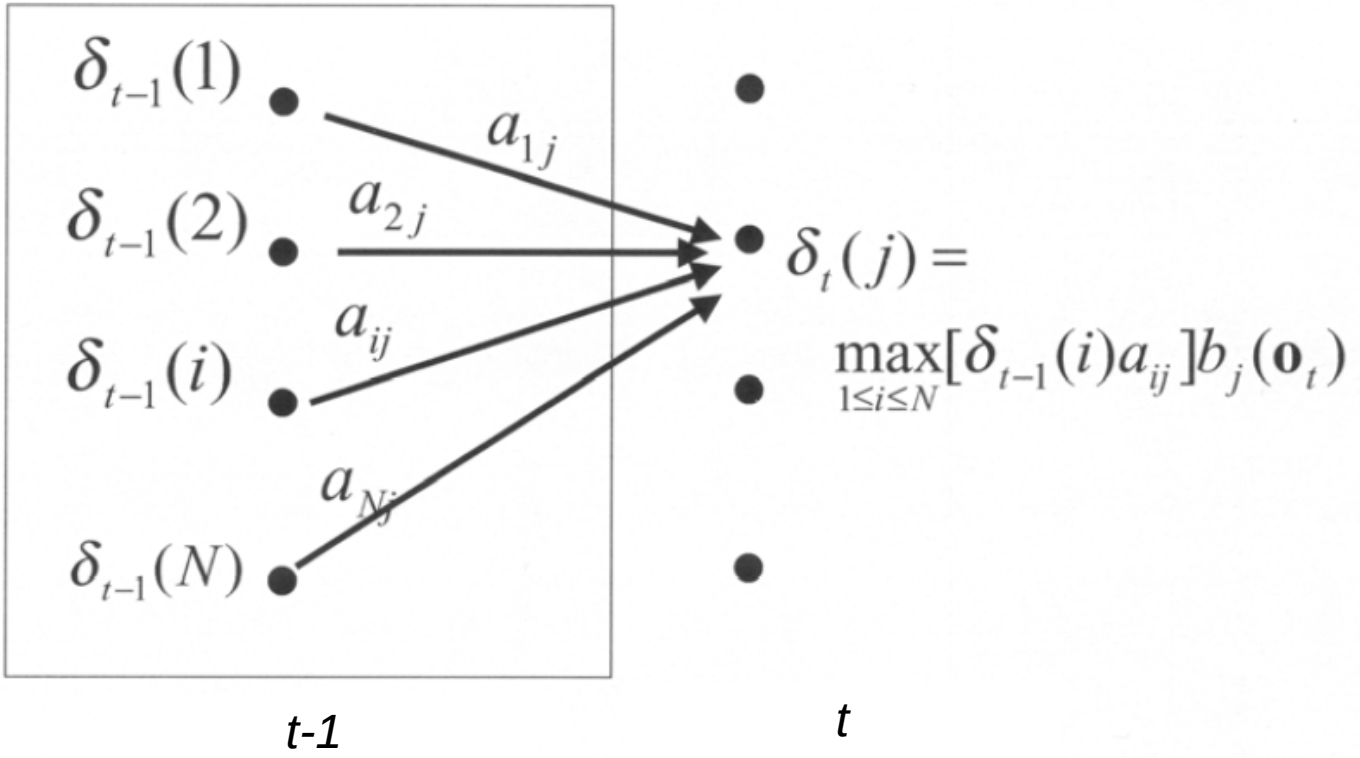
$$\delta_t(j) = \max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}] b_j(o_t)$$

$$\psi_t(j) = \arg\max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}]$$

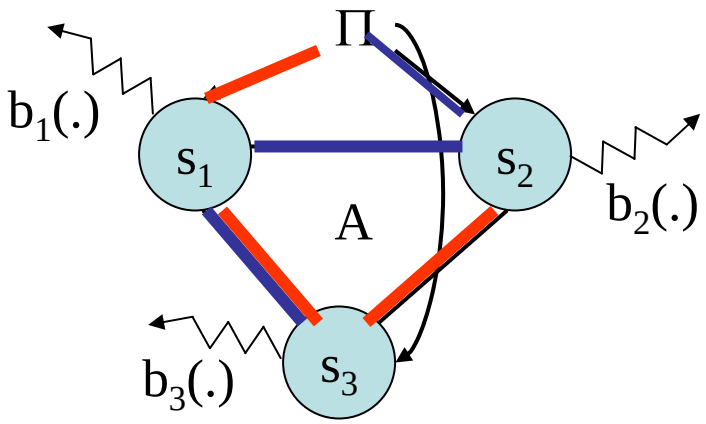
Terminaison $P^* = \max_{1 \le i \le N}[\delta_T(i)] \qquad q_T^* = \arg\max_{1 \le i \le N}[\delta_T(i)]$

Suite d'états retenue $q_t^* = \psi_{t+1}(q_{t+1}^*)$

Exemple Algorithme de Viterbi



Algorithme de Viterbi (suite)



$$\max_{I} P(O, I|H)$$

Recherche parmi tous les chemins possibles (m^n)

~ problèmes de recherche de chemins min. (max.) dans un graphe

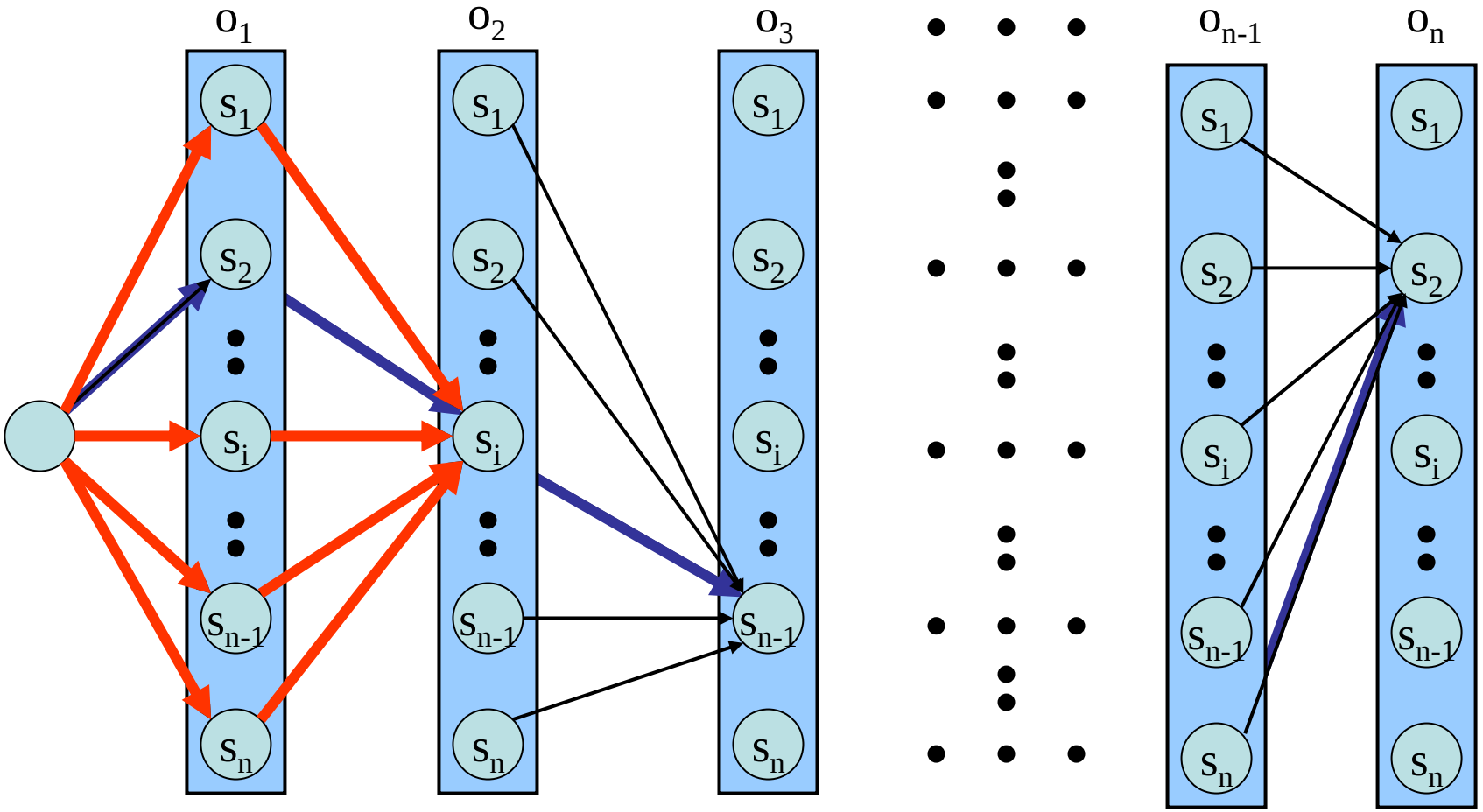
approches gloutonnes (Ex.: Dijkstra, Kruskall,etc...), programmation dynamique

Viterbi ~ Prog. dynamique dans un treillis particulier

Algorithme de Viterbi(suite)

De G à D avec $d(s_{i,t},s_{j,t+1})=a_{i,j}*b_j(o_{t+1})$

Principe d'optimalité de Bellman



Problèmes de base des HMM

1. Évaluation:
   1. Problème: calculer la probabilité d'observation de la séquence d'observation étant donnée un HMM: $P(O \mid \lambda)$
   2. Solution: Forward Algorithm
2. Décodage:
   1. Problème: trouver la séquence d'état qui maximise la séquence d'observations
   2. Solution: Viterbi Algorithm
3. Entraînement:
   1. Problème: ajuster les paramètres du modèle HMM afin de maximiser la probabilité de générer une séquence d'observation à partir de données d'entraînement
   2. Solution: Forward-Backward Algorithm

Apprentissage.

Le but de l'apprentissage est de déterminer les paramètres (A,B,π) qui maximisent la probabilité de la suite d'observations $P(O \mid \lambda)$.

L'idée employée ici est d'utiliser des procédures de **ré-estimation** par punition-récompense :

- choisir un ensemble initial de paramètres λ_0 ;
- calculer λ_1 à partir de λ_0 ;
- répéter ce processus jusqu'à un critère de fin.

Partant de λ_n, λ_{n+1} doit vérifier :

$$\prod_r P(O^r \mid \lambda_{n+1}) \geq \prod_r P(O^r \mid \lambda_n)$$

On cherche donc à définir une fonction F telle que $\lambda_{n+1} = F(\lambda_n)$.

Apprentissage.

L'approche la plus simple pour définir F consiste à faire des statistiques sur l'utilisation des transitions et des distributions. Ceci revient à calculer des fréquences d'utilisation à partir de l'ensemble d'apprentissage.

Si l'ensemble est important, ces fréquences fournissent une bonne approximation des probabilités **a posteriori** utilisables alors comme paramètres du modèle pour l'itération suivante.

La méthode d'apprentissage va donc consister à partir d'un modèle initial aléatoire, à estimer ses paramètres comme indiqué ci-dessus, puis à recommencer cette estimation ("ré-estimation") jusqu'à obtenir une certaine convergence.

n est le nombre d'états du HMM, N le nombre de séquences d'apprentissage.

Les formules de réestimation

On définit $\xi_t^k(i,j)$ comme la probabilité, étant donnés une phrase O^k et un HMM Λ, que ce soit l'état s_i qui ait émis la lettre de rang t de O^k et l'état s_j qui ait émis celle de rang $t+1$. Donc :

$$\xi_t^k(i,j) = P(q_t = s_i,\, q_{t+1} = s_j \mid O^k,\Lambda)$$

Ce qui se réécrit :

$$\xi_t^k(i,j) = \frac{P(q_t = s_i,\, q_{t+1} = s_j,\, O^k \mid \Lambda)}{P(O^k \mid \Lambda)}$$

Par définition des fonctions *forward-backward*, on en déduit :

$$\xi_t^k(i,j) = \frac{\alpha_t^k(i) a_{ij} b_j(O_{t+1}^k) \beta_{t+1}^k(j)}{P(O^k \mid \Lambda)}$$

Les formules de réestimation

On définit aussi la quantité $\gamma_t^k(i)$ comme la probabilité que la lettre de rang t de la phrase O^k soit émise par l'état s_j.

$$\gamma_t^k(i) = P(q_t = s_i \mid O^k,\Lambda)$$

Soit :

$$\gamma_t^k(i) = \sum_{j=1}^{n} P(q_t = s_i,\ q_{t+1} = s_j \mid O^k,\Lambda) = \frac{\sum_{j=1}^{n} P(q_t = s_i,\ q_{t+1} = s_j, O^k \mid \Lambda)}{P(O^k \mid \Lambda)}$$

On a la relation :

$$\gamma_t^k(i) = \sum_{j=1}^{n} \xi_t(i,j) = \frac{\alpha_t^k(i)\beta_t^k(i)}{P(O^k \mid \Lambda)}$$

Les formules de réestimation

Le nouveau modèle HMM se calcule à partir de l'ancien en réestimant π, A et B par comptage sur la base d'apprentissage. On mesure les fréquences :

$\overline{\pi}_i$ = nombre de fois que HMM s'est trouvé dans s_i à l'instant 1

$$\overline{a}_{ij} = \frac{\text{nombre de fois où la transition de } s_i \text{ à } s_j \text{ a été utilisée}}{\text{nombre de transitions effectuées à partir de } s_i}$$

$$\overline{b}_j(k) = \frac{\text{nombre de fois où le HMM s'est trouvé dans l'état } s_j \text{ en observant } v_k}{\text{nombre de fois où le HMM s'est trouvé dans l'état } s_j}$$

Les formules de réestimation

$$\overline{\pi}_i = \frac{1}{N} \sum_{k=1}^{N} \gamma_1^k(i) \qquad \overline{a}_{ij} = \frac{\sum_{k=1}^{N} \left(\sum_{t=1}^{|O^k|-1} \xi_t^k(i,j) \right)}{\sum_{k=1}^{N} \sum_{t=1}^{|O^k|-1} \gamma_t^k(i)}$$

$$\overline{b}_j(l) = \frac{\sum_{k=1}^{N} \left(\sum_{t=1,|O^k|-1 \text{ avec } O_t^k = v_l} \gamma_t^k(j) \right)}{\sum_{k=1}^{N} \sum_{t=1}^{|O^k|-1} \gamma_t^k(j)}$$

Algorithme de Baum-Welch

❶ Fixer les valeurs initiales :

a_{ij}^0, $b_j^0(k)$, π_i^0 $\quad 1 \leq i,j \leq n, \quad 1 \leq k \leq n.$

❷ Calculer à l'aide des fonctions Forward-Backward :

$\xi(i,j)$, $\gamma_t(i)$ $\quad 1 \leq i,j \leq n \quad 1 \leq t \leq T-1$

et $\overline{\lambda}$ en utilisant les formules de ré-estimation.

❸ Recommencer en 2 jusqu'à ce qu'un certain critère de convergence soit rempli.

Un exemple : 1

On part du HMM Λ_0 défini par les paramètres suivants :

$$A = \begin{pmatrix} 0.45 & 0.35 & 0.20 \\ 0.10 & 0.50 & 0.40 \\ 0.15 & 0.25 & 0.60 \end{pmatrix} \quad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.5 & 0.5 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.5 \\ 0.3 \\ 0.2 \end{pmatrix}$$

On peut calculer sur l'alphabet à deux lettres $\{a,b\}$:

$$P(a\ b\ b\ a\ a \mid \Lambda_0) = 0.0278$$

Un exemple : 2

Si on prend comme ensemble d'apprentissage cette seule phrase, l'application de l'algorithme de Baum-Welsh doit augmenter sa probabilité de reconnaissance.

Après une réestimation, on trouve Λ_1:

$$A = \begin{pmatrix} 0.346 & 0.365 & 0.289 \\ 0.159 & 0.514 & 0.327 \\ 0.377 & 0.259 & 0.364 \end{pmatrix} \quad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.631 & 0.369 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.656 \\ 0.344 \\ 0.0 \end{pmatrix}$$

$$P(a\ b\ b\ a\ a \mid \Lambda_1) = 0.0529$$

Un exemple: 3

Après 15 itérations :

$$A = \begin{pmatrix} 0.0 & 0.0 & 1.0 \\ 0.212 & 0.788 & 0.0 \\ 0.0 & 0.515 & 0.485 \end{pmatrix} \quad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.969 & 0.031 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$$

$$P(a\ b\ b\ a\ a \mid \Lambda_{15}) = 0.2474$$

Remarques :

- Le choix du modèle initial influe sur les résultats ; par exemple, si certaines valeurs de A et B sont égales à 0 au départ, elles le resteront jusqu‘à la fin de l'apprentissage. Ceci permet en particulier de garder la structure dans les modèles gauches-droits.
- L'algorithme converge vers des valeurs de paramètres qui assurent un maximum local de P(O/λ). Il est donc important, si l'on veut être aussi prés que possible du minimum global, de bien choisir la structure et l'initialisation.
- Le nombre d'itérations est fixé empiriquement. L'expérience prouve que, si le point précédent a été correctement traité, la stabilisation des paramètres ne correspond pas à un sur apprentissage : il n'y a donc en général pas besoin de contrôler la convergence par un ensemble de validation. Mais cette possibilité est évidemment toujours a disposition.

Les techniques à base de HMM sont à l'heure actuelle des plus utilisées, le modèle HMM présente différentes avantages: clarté, rigueur et généralité. Des modèles hybrides ont été proposés afin de combiner les HMM avec d'autres modèles telles que les approches connexionnistes dans le cadre de l'intelligence artificielle.

Avantages	**Inconvénients**
•Base mathématique solide pour comprendre son fonctionnement. •Variabilité de la forme. •Alignement temporel incorporé systématiquement. •Prise en compte de l'ordre d'apparition dans une séquence de vecteurs. •Reconnaissance réalisée par un simple calcul de probabilité cumulée. •Décision globale sans obligation d'utiliser des seuils. •Séparation franche entre données et algorithmes.	•Ignorance complète de la durée relative des événements du signal. •Dégradation des performances si l'apprentissage n'est pas suffisant. •Le choix à priori de la typologie des modèles(nombre d'états, transitions autorisées et règles de transition) limite la souplesse des modèles. •Modèle comportemental et non fonctionnel.