Introduction aux modèles de Markov

Philippe Langlais
felipe@iro.umontreal.ca

February 2, 2015

Plan

Modèles visibles versus cachés

Problème 1: calcul de la probabilité d'une observation

Problème 2: recherche de la séquence d'états

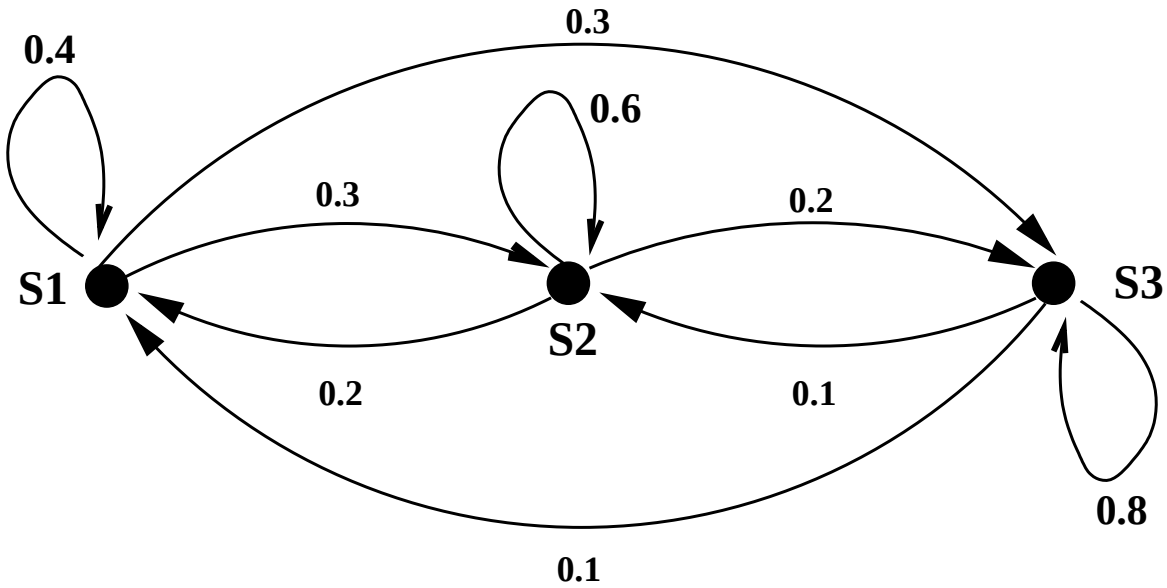
Problème 3: apprendre λ

HMMs en reconnaissance de la parole

Modèle de Markov visible

À propos du temps

$S_1 = \text{pluie}; S_2 = \text{nuage}; S_3 = \text{soleil}$



$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$ matrice de transitions

Note: $\forall (i,j) \in [1,3],\ a_{ij} >= 0,$ et $\forall i \in [1,3], \sum_{j=1}^{3} a_{ij} = 1$

Quel temps fera-t-il les huit prochains jours ?

$$p(S_3 S_3 S_3 S_1 S_1 S_3 S_2 S_3) \stackrel{def}{=}$$

$$p(S_3)p(S_3|S_3)p(S_3|S_3 S_3)p(S_1|S_3 S_3 S_3)p(S_1|S_1 S_3 S_3 S_3)$$
$$p(S_3|S_1 S_1 S_3 S_3 S_3)p(S_2|S_3 S_1 S_1 S_3 S_3 S_3)$$
$$p(S_3|S_2 S_3 S_1 S_1 S_3 S_3 S_3) \approx$$

$$p(S_3)p(S_3|S_3)p(S_3|S_3)p(S_1|S_3)p(S_1|S_1)$$
$$p(S_3|S_1)p(S_2|S_3)p(S_3|S_2) =$$
$$\pi_3 \times a_{33} \times a_{33} \times a_{31} \times a_{11} \times a_{13} \times a_{32} \times a_{23} = 1.536 \times 10^{-4}$$

- hypothèse markovienne d'ordre d'ordre n: $p(S_k|S_{k-1} \ldots S_1) = p(S_k|S_{k-1} \ldots S_{k-n})$
- indépendance au temps: $p(q_t = S_j|q_{t-1} = S_i) = p(S_j|S_i)$

Combien de temps va-t-il pleuvoir ?

- $p_i(d) = \sum_{j \neq i} p(\overbrace{S_i S_i \ldots S_i}^{d} S_j)$ avec $j \neq i$
  $= a_{ii}^{d-1}(1 - a_{ii})$ exponentiel

- Espérance d'avoir d jours le même temps (s_i):

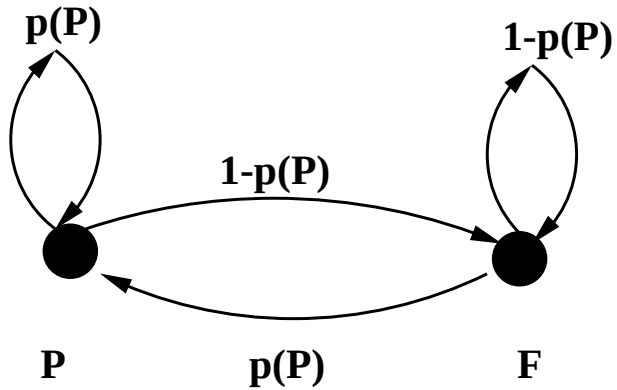
$$E[d] = \sum_{d=1}^{\infty} d p_i(d) = \sum_{d=1}^{\infty} d a_{ii}^{d-1}(1 - a_{ii}) = \frac{1}{1 - a_{ii}}$$

rappel: $\sum_{d=1}^{\infty} d a^d = \frac{a}{(1-a)^2}$ avec $-1 < a < 1$

- Donc l'espérance du nombre de jours où:
  - il fera beau est $1/(1 - 0.8) = 5$;
  - où le temps sera nuageux est $1/(1 - 0.6) = 2.5$;
  - où il pleuvra: $1/(1 - 0.4) = 1.67$

Modèles markoviens et jets de pièces

- On vous annonce oralement le résultat de tirages (pile ou face) sans vous montrer comment on procède aux tirages.
- Première modélisation: Il existe une pièce (possiblement biaisée). Un état pour pile, un état pour face. Avec ce modèle chaque observation spécifie la séquence d'états
  - c'est un modèle visible (VMM)

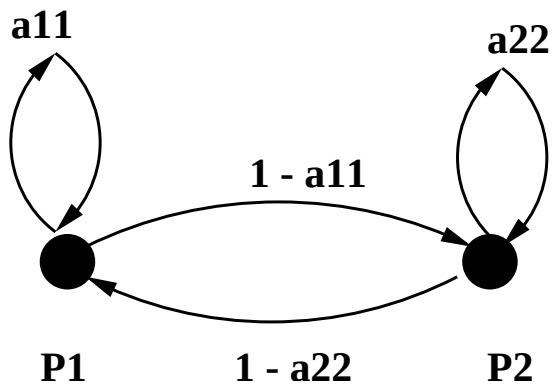


O	=	P	P	F	F	P	F	F	P	...
S	=	1	1	2	2	1	2	2	1	...

paramètre: $p(Pile)$

Modèles markoviens et jets de pièces

- Deuxième modèlisation: Il existe deux pièces (possiblement biaisées de manière différente) que l'opérateur change à son gré lors de chaque tirage.
- Cette fois-ci, l'observation des tirages ne nous spécifie pas dans quel état (pièce) de notre modèle on se trouve.
  - la séquence d'états S est cachée.

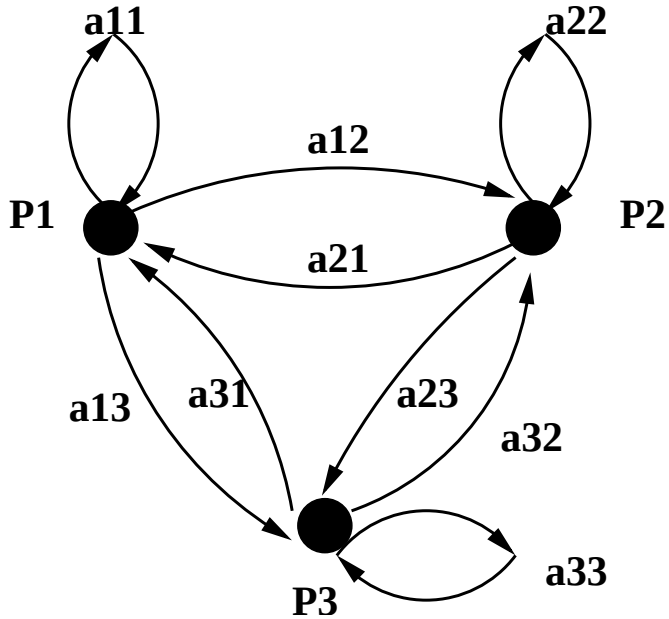


O	=	P	P	F	F	P	F	F	P	...
S	=	1	2	1	2	1	2	2	1	...

4 paramètres: $p_1(Pile), p_2(Pile), a_{11}, a_{22}$

Modèles markoviens et jets de pièces

- Troisième modélisation: Il existe trois pièces (possiblement biaisées de manière différente) que l'opérateur change à son gré lors de chaque tirage.



O	=	P	P	F	F	P	F	F	P	...
S	=	1	3	2	3	1	3	2	1	...

9 paramètres: $p_{1,2,3}(Pile), \{a_{ij}\}, \forall i \in [1,3], j \in [2,3]$

Analogie

Des urnes et des balles[1]

- N urnes qui contiennent chacune des balles de couleur. Il y a M couleurs différentes de balles.
- Processus génératif: Un génie tire une balle d'une urne initiale. Il annonce la couleur de la balle et repose la balle dans son urne. Selon un processus aléatoire dépendant de la dernière urne concernée, le génie choisit une nouvelle urne (qui peut-être la même que la précédente) et effectue le tirage d'une balle dans cette urne, etc.
- Modélisation possible: un HMM à N états (un par urne); chaque état émet une balle de couleur selon une loi de probabilité spécifique à l'état. Le choix d'une urne par le génie est modélisé par les probabilités de transition d'un état vers un autre.

[1] Analogie proposée par Jack Ferguson

Des urnes et des balles

- 3 couleurs: V J R (symboles), 3 urnes (états: s_1, s_2, s_3).
- Le modèle est spécifié par $\lambda = \{\pi, A, B\}$ où $\pi = \{0.3, 0.3, 0.4\}$, et

$$A = \left[\begin{array}{c|ccc} & s_1 & s_2 & s_3 \\ \hline s_1 & 0.5 & 0.3 & 0.2 \\ s_2 & 0.4 & 0.0 & 0.6 \\ s_3 & 0.0 & 0.3 & 0.7 \end{array}\right] \quad B = \left[\begin{array}{c|ccc} & V & J & R \\ \hline s_1 & 0.0 & 1.0 & 0.0 \\ s_2 & 0.5 & 0.1 & 0.4 \\ s_3 & 0.2 & 0.0 & 0.8 \end{array}\right]$$

- A et B sont respectivement les matrices de transition et d'émission.
- Soit une séquence d'observations: $O = \{VJRV\}$ $p(O|\lambda)$?

Des urnes et des balles

- Il existe 5 chemins (séquences) qui génèrent O:

$$c_1 = \{s_2 s_1 s_3 s_3\} \quad c_3 = \{s_2 s_1 s_2 s_3\} \quad c_5 = \{s_3 s_2 s_3 s_3\}$$
$$c_2 = \{s_2 s_1 s_3 s_2\} \quad c_4 = \{s_3 s_2 s_3 s_2\}$$

$$\begin{aligned}
p(O, c_1|\lambda) &= \pi_2.b_2(V).a_{21}.b_1(J).a_{13}.b_3(R).a_{33}.b_3(V) &&= 0.00134 \\
p(O, c_2|\lambda) &= \pi_2.b_2(V).a_{21}.b_1(J).a_{13}.b_3(R).a_{32}.b_2(V) &&= 0.00144 \\
p(O, c_3|\lambda) &= \pi_2.b_2(V).a_{21}.b_1(J).a_{12}.b_2(R).a_{23}.b_3(V) &&= 0.000664 \\
p(O, c_4|\lambda) &= \pi_3.b_3(V).a_{32}.b_2(J).a_{23}.b_3(R).a_{32}.b_2(V) &&= 0.0001728 \\
p(O, c_5|\lambda) &= \pi_3.b_3(V).a_{32}.b_2(J).a_{23}.b_3(R).a_{33}.b_3(V) &&= 0.00016128 \\
\hline
p(O|\lambda) &= \textstyle\sum_{i=1}^{5} p(O, c_i, |\lambda) \times p(c_i|\lambda) &&= 0.00377808
\end{aligned}$$

- La séquence qui explique le mieux l'observation est c_2
- **Note:** les calculs peuvent être factorisés

Caractérisation d'un HMM

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

les états ($\{S_1, \ldots, S_N\}$): la séquence d'états est cachée, mais un état correspond souvent à un phénomène précis (ex. urne).

les observations différentes ($\{v_1, \ldots, v_M\}$): l'alphabet avec lequel on décrit les observations (ex: la couleur des balles).

les probabilités de transition ($A = \{a_{i,j}\}$) pour tout $(i,j) \in [1,N]$ avec: $a_{ij} = p(q_t = S_j | q_{t-1} = s_i), \ \forall (i,j) \in [1,N]$

Note: $a_{i,j} \geq 0$ et $\sum_{j=1}^{N} a_{ij} = 1 \ \forall (i,j) \in [1,N]$

les probabilités d'émission ($B = \{b_j(k)\}$) pour tout $j \in [1,N]$ et $k \in [1,M]$ avec: $b_j(k) = p(v_k \text{ à l'instant } t | q_t = j)$

Note: $b_j(k) \geq 0$ et $\sum_{o=1}^{M} b_j(o) = 1$

les probabilités initiales ($\pi = \{\pi_i\}$) avec $\pi_i = p(q_1 = S_i), \ \forall i$

Note: $\pi_i(k) \geq 0$ et $\sum_{i=1}^{N} \pi_i = 1$

Les trois problèmes fondamentaux des HMMs

Évaluation: Sachant $O = \{O_1 O_2 \dots O_T\}$ et $\lambda = (A, B, \pi)$, comment calculer $p(O|\lambda)$?
Évaluer une observation selon un modèle

Retirer le H de Hidden: Sachant $O = \{O_1 O_2 \dots O_T\}$ et $\lambda = (A, B, \pi)$, comment trouver la séquence (cachée) **optimale** d'états?
Ce qui en pratique nous intéresse le plus souvent

Apprentissage: Sachant un corpus d'entraînement O, comment ajuster les paramètres λ du modèle ?
Le problème le plus difficile

Solution au problème 1

Évaluation

- Soit $Q = q_1 q_2 \dots q_T$ une séquence d'états pouvant "expliquer" O.

$$p(O|\lambda) = \sum_{\text{all } Q} p(O, Q|\lambda) = \sum_{\text{all } Q} p(O|Q, \lambda) p(Q|\lambda)$$

- avec:
  $p(O|Q, \lambda) = \prod_{t=1}^{T} p(o_t|q_t, \lambda) = b_{q_1}(o_1) \times b_{q_2}(o_2) \dots b_{q_T}(o_T)$
  $p(Q|\lambda) = \pi_{q_1} \times a_{q_1 q_2} \times \dots a_{q_{T-1} q_T}$
- donc:

$$p(O|\lambda) = \sum_{q_1 \dots q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \dots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

Complexité: $(2T-1) \times N^T$ multiplications, $N^T - 1$ additions
Ex: $N = 5$ (états), $T = 100$ (observations)
$\triangleright$ de l'ordre de $2 \times 100 \times 5^{100} \approx 10^{72}$ opérations !

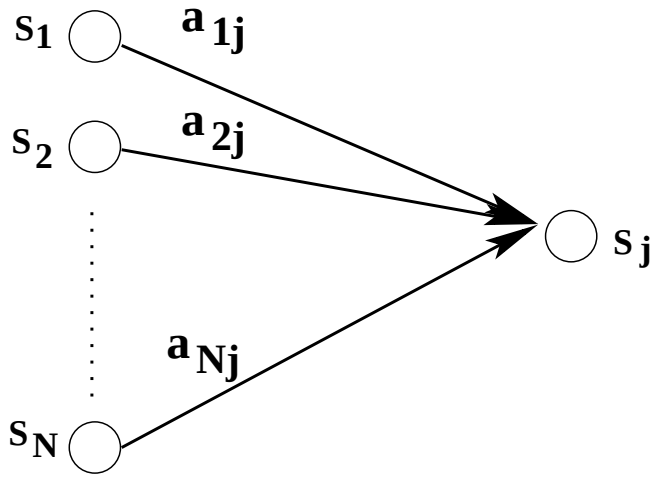
Solution au problème 1

Calcul forward (en avant): $\alpha_t(i) = p(o_1 \dots o_t, q_t = s_i|\lambda)$

Init: $\alpha_1(i) = \pi_i b_i(o_1), \ \forall i \in [1, N]$

Induction: $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1})$, pour tout $t \in [1, T-1]$ et pour tout $j \in [1, N]$

Terminaison: $p(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$



Complexité: $o(N^2 \times T)$ opérations au lieu de $o(T \times N^T)$

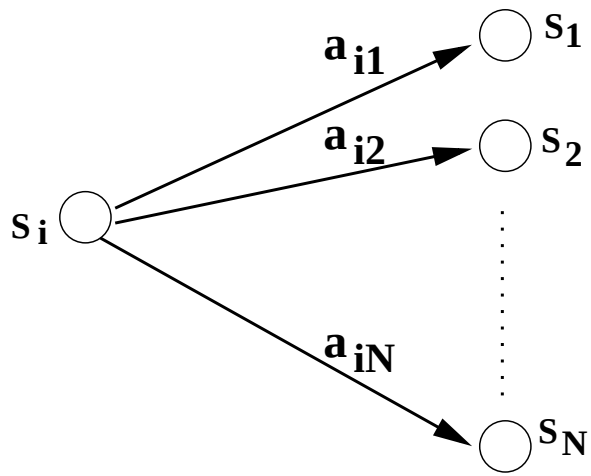
Ex: $N = 5$, $T = 100 \triangleright$ environ 3000 opérations (*vs* 10^{72} !!!)

Solution au problème 1

Calcul backward (en arrière): $\beta_t(i) = p(o_{t+1} \dots o_T | q_t = s_i, \lambda)$

Init: $\beta_T(i) = 1, \ \forall i \in [1, N]$

Induction: $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$ pour tout $t \in [1, T-1]$ et pour tout $i \in [1, N]$



Note: même complexité que le calcul forward.

Solution au problème 2 (séquence optimale)

$\gamma_t(i) = p(q_t = s_i|O, \lambda)$

- Pas de réponse unique.
- **Un critère possible:** choisir l'état le plus probable individuellement pour chaque t:

$$\hat{q}_t = \underset{1 \leq i \leq N}{\operatorname{argmax}}[\gamma_t(i)], \ \forall t \in [1, T]$$

  - où $\gamma_t(i)$ la probabilité d'être dans l'état s_i au temps t:
    $\gamma_t(i) = \frac{p(q_t = s_i, O|\lambda)}{p(O|\lambda)} = \frac{\alpha_t(i) \times \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \times \beta_t(i)}$

  (On peut calculer les γ une fois les α et β calculés)

- **Problème:** rien ne garantit que les transitions entre chaque état de $\hat{Q}$ sont valides
- Critère local.

Solution au problème 2 (séquence optimale)

Algorithme de Viterbi

- $\hat{Q} = \text{argmax}_Q \; p(Q|O,\lambda) = \text{argmax}_Q \, p(O,Q|\lambda)$
  car $p(O,Q|\lambda) = p(Q|O,\lambda) \times p(O|\lambda)$
- Posons:

$$\delta_t(i) = \max_{q_1 \ldots q_{t-1}} p(q_1 q_2 \ldots q_t = s_i, o_1 \ldots o_t|\lambda)$$

- Par induction:

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

- On conserve pour chaque t et chaque i l'état ayant amené au maximum $\delta_t(j)$, soit $\phi_t(j)$

Solution au problème 2

Viterbi

init: $\delta_1(i) = \pi_i b_i(o_1)$ et $\phi_1(i) = 0$

récursion:

$$\begin{aligned} \delta_t(j) &= \max_{1 \le i \le N}[\delta_{t-1}(i)a_{ij}]b_j(o_t) \quad 2 \le t \le T \\ \phi_t(j) &= \operatorname{argmax}_{1 \le i \le N}[\delta_{t-1}(i)a_{ij}] \quad 1 \le j \le N \end{aligned}$$

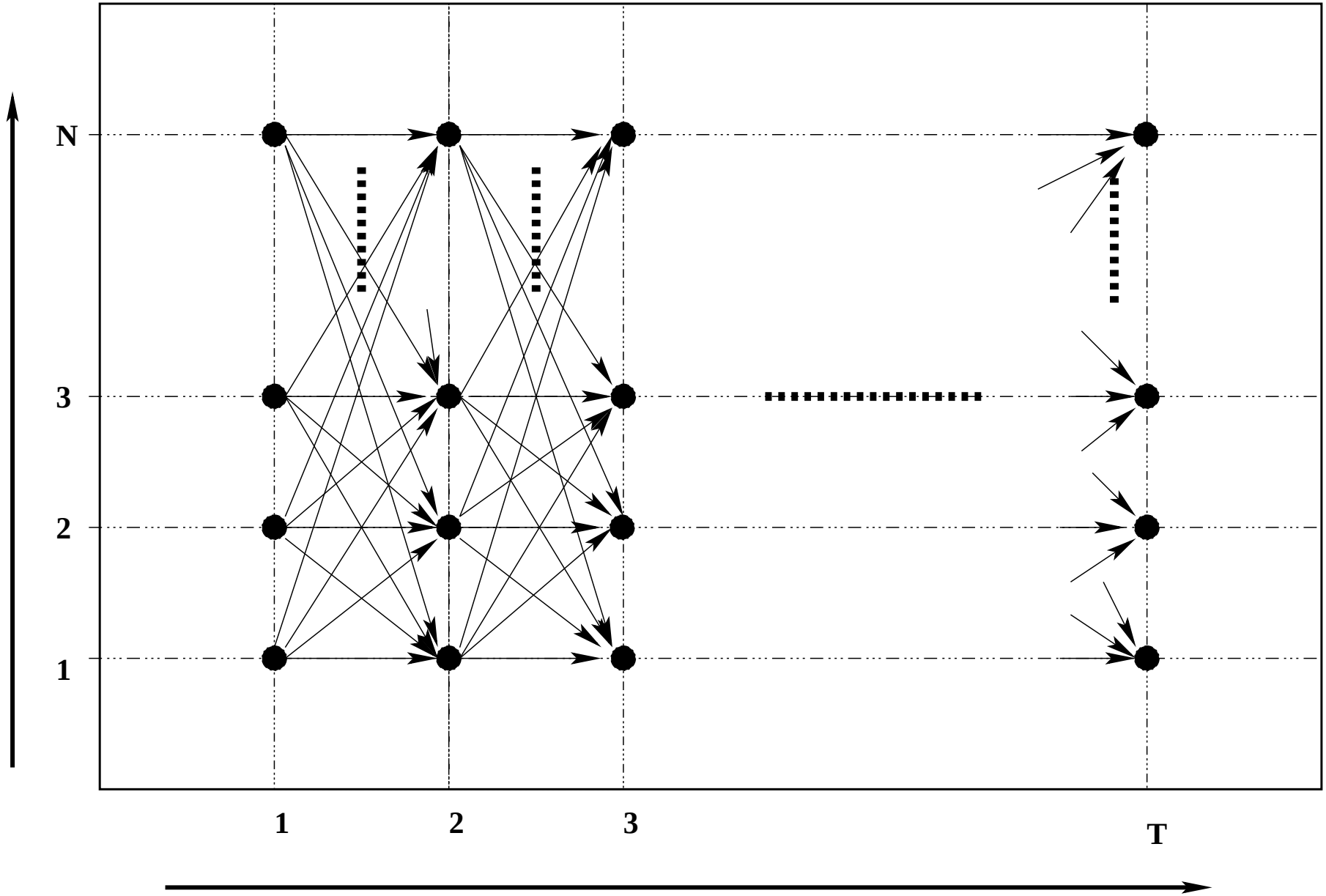
fin:

$$\hat{p} = \max_{1 \le i \le N} \delta_T(i)$$

$$\hat{q}_T = \operatorname*{argmax}_{1 \le i \le N} \delta_T(i)$$

meilleure séquence: $\hat{q}_t = \phi_{t+1}(\hat{q}_{t+1}),\ t = T-1, T-2, \ldots, 1$

Structure en treillis



Algorithme de Viterbi

code

- Soit S une table $N \times T$, telle que:
  $S[t,i] = (\gamma_t(i), \phi_t(i)) = (S[t,i].p, S[t,i].b)$

- Soit $A[i,j]$ la matrice de transition et $B[i,k]$ la matrice d'émission, avec $(i,j) \in [1,N]$ et $k \in [1,M]$.
  (matrice de transition souvent creuse)

- Pour éviter de faire un cas particulier pour les probabilités initiales, on peut étendre A d'une ligne et d'une colonne:
  $A[0,i] = \pi_i$ et $A[i,0] = 0 \ \forall i \in [1,N]$

Algorithme de Viterbi

code

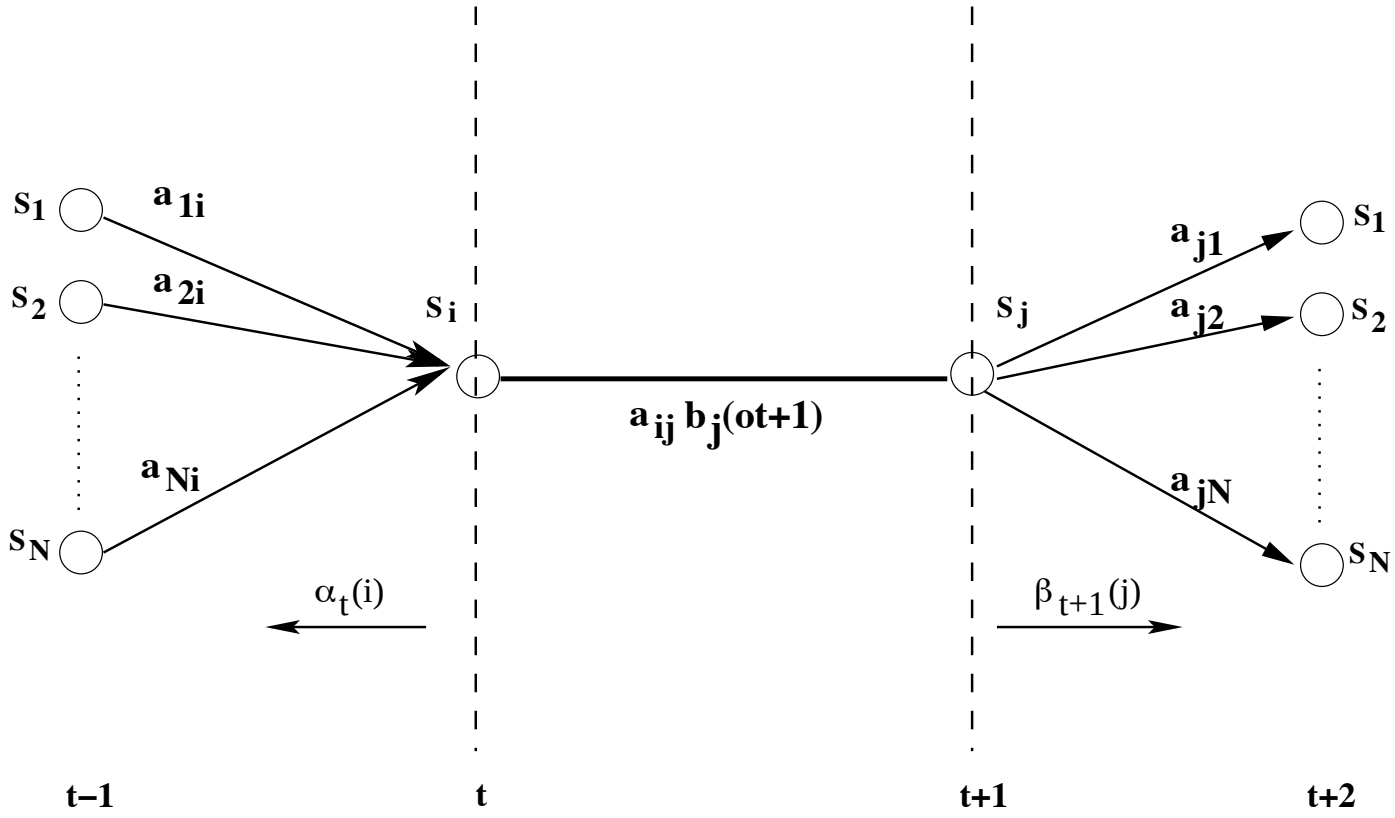
// Décodage de $o_1 o_2 \dots o_T$ (à faire dans le domaine des log !)
$S[0,0].p \leftarrow 1$
for t $\leftarrow$ 1 à T **do**
  for i $\leftarrow$ 1 à N **do**
    $S[t,i].p \leftarrow 0$
    $e \leftarrow B[i, o_t]$
    for j $\leftarrow$ 1 à N **do**
      if $(A[j,i] > 0)$ **et** $((m = (S[t-1,j].p \times A[j,i])) > S[t,i].p)$ **then**

        $S[t,i] \leftarrow (m,j)$
    $S[t,i].p \leftarrow S[t,i].p \times e$
// Retour
$\hat{s} \leftarrow \text{argmax}_{i \in [1,N]: S[T,i].p > 0} S[T,i].p$
if $S[T,\hat{s}].p > 0$ **then**
  Retourner le chemin à partir de $S[T,\hat{s}]$
else
  Échec de reconnaissance

Problème 3 (estimer λ)

Algorithme Baum-Welch

- à maximum de vraisemblance, par une procédure itérative (EM) qui maximise $p(O|\lambda)$
- Soit $\xi_t(i,j) = p(q_t = s_i, q_{t+1} = s_j|O,\lambda)$ la probabilité de transiter de i vers j au temps t sachant l'observation O et le modèle.



Algorithme Baum-Welch

$$\xi_t(i,j) = \frac{p(q_t=s_i, q_{t+1}=s_j, O|\lambda)}{p(O|\lambda)}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}$$

Note: $\gamma_t(i) = p(q_t = s_i|O,\lambda)$, d'où: $\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$

- $\sum_{t=1}^{T-1} \gamma_t(i) =$ le nombre espéré de transitions depuis s_i, sachant l'observation O et le modèle.
- $\sum_{t=1}^{T-1} \xi_t(i,j) =$ le nombre espéré de transitions depuis s_i vers s_j, sachant O et le modèle.

Algorithme Baum-Welch

Avec un peu (beaucoup?) d'intuition

$\overline{\pi_i}$ = nombre espéré de fois où au temps 1 on est en s_i $(= \gamma_1(i))$

$$\overline{a_{ij}} = \frac{\text{nb. espéré de transitions de } s_i \text{ vers } s_j}{\text{nb. espéré de transitions depuis } s_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\overline{b_j(k)} = \frac{\text{nb. espéré de fois où on est en } s_j \text{ et on observe } v_k}{\text{nb. espéré de fois où on est dans } s_j} = \frac{\sum_{t=1:o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

- **Baum72** démontre la convergence de cet algorithme

Algorithme Baum-Welch

Une instance de EM

- Variable cachée: la séquence d'états q
- La fonction auxiliaire A est:

$$A(\lambda, \lambda') = \sum_{q \in \mathcal{Q}} p(O, q|\lambda') \log p(O, q|\lambda)$$

ou (ce qui revient au même en terme de maximisation):

$$A'(\lambda, \lambda') = \sum_{q \in \mathcal{Q}} p(q|O, \lambda') \log p(O, q|\lambda)$$

Algorithme Baum-Welch

Une instance de EM

- Posons $q = q_1 \dots q_T$ et $O = o_1 \dots o_T$.
- $p(O, q|\lambda) = \pi_{q_1} b_{q_1}(o_1) \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t}(o_t)$

- On peut décomposer A en 3 termes indépendants (au regard de la maximisation):

$$
\begin{aligned}
&A(\lambda, \lambda') = \\
&\sum_{q \in \mathcal{Q}} (\log \pi_{q_1} \times p(O, q|\lambda') + \log b_{q_1}(o_1) \times p(O, q|\lambda')) + \\
&\sum_{q \in \mathcal{Q}} \left(\sum_{t=2}^{T} \log a_{q_{t-1} q_t} \right) p(O, q|\lambda') + \\
&\sum_{q \in \mathcal{Q}} \left(\sum_{t=2}^{T} \log b_{q_t}(o_t) \right) p(O, q|\lambda')
\end{aligned}
$$

Algorithme Baum-Welch

Une instance de EM: cas des π_i

- Condition nécessaire de la maximisation: $\frac{\delta}{\delta \pi_i} A = 0$
- Maximiser A (selon π_i) est équivalent à:
  - maximiser seulement $\sum_{q \in \mathcal{Q}} \log \pi_{q_1} p(O, q|\lambda')$
  - qui revient à maximiser $\sum_{i=1}^{N} \log \pi_i p(O, q_1 = i|\lambda')$

- Coefficient de Lagrange (μ) pour la contrainte $\sum_{j=1}^{N} \pi_j = 1$:

$$\frac{\delta}{\delta \pi_i} \left(\sum_{i=1}^{N} \log \pi_i p(O, q_1 = i|\lambda') - \mu \left(\sum_{j=1}^{N} \pi_j - 1 \right) \right) = 0$$

Algorithme Baum-Welch

Une instance de EM: cas des π_i

$$\frac{p(O, q_1 = i|\lambda')}{\pi_i} - \mu = 0 \; \forall i \in [1, N]$$

Soit:

$$\pi_i = \frac{p(O, q_1 = i|\lambda')}{\mu} \; \forall i \in [1, N]$$

Or:

$$\sum_{i=1}^{N} \pi_i = 1 = \sum_{i=1}^{N} \frac{p(O, q_1 = i|\lambda')}{\mu} \Longrightarrow \mu = \sum_{i=1}^{N} p(O, q_1 = i|\lambda')$$

D'où :

$$\pi_i = \frac{p(O, q_1 = i|\lambda')}{\sum_i p(O, q_1 = i|\lambda')} = \gamma_1(i)$$

Limitations de l'approche à maximum de vraisemblance

- Soit V modèles (λ_v), $v \in [1, V]$, et une tâche de reconnaissance:

$$\hat{v} = \underset{v}{\operatorname{argmax}}\, p(O|\lambda_v)$$

- Lors de l'entraînement ML des modèles, on a:

$$p_v^\star = \max_{\lambda_v} p(O^v|\lambda_v)$$

où O^v représente l'ensemble des données étiquetées v dans le corpus d'entraînement (O).

- un jeu d'observations séparé pour l'entraînement de chaque modèle

Autre critère

Maximum Mutual Information (MMI)

$$I = \max_{\lambda} \left\{ \sum_{v=1}^{V} \left[\log p(O^v|\lambda_v) - \log \sum_{k \neq v} p(O^v|\lambda_k) \right] \right\}$$

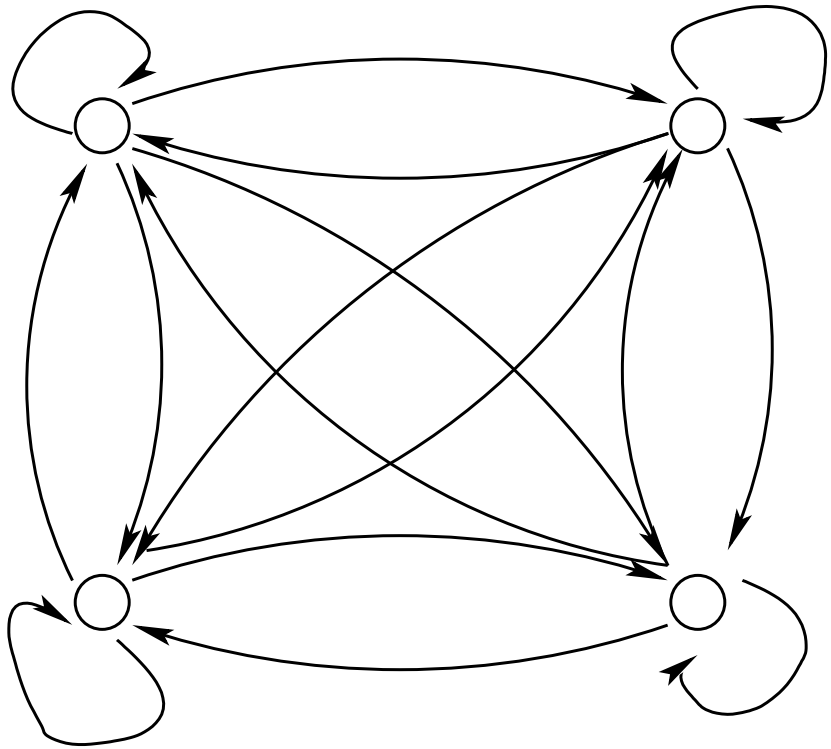
avec $\lambda = \{\lambda_1, \ldots, \lambda_V\}$

- Apprentissage plus coûteux

Topologie d'un modèle

Modèles ergodiques

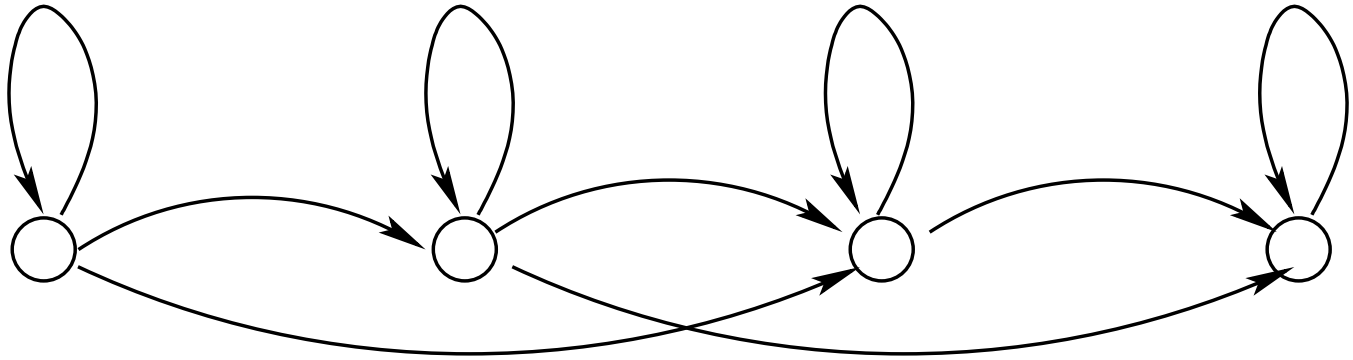
- Un modèle est ergodique si tout état est atteignable depuis tout autre état en un nombre fini de transitions.
- Exemple pour $N = 4$:



Topologie d'un modèle

Modèles gauche-droite

- modèle de Bakis:



- Dans un modèle gauche-droite: $a_{ij} = 0$ si $j < i$

$$\pi_i = \begin{cases} 1 & \text{si } i = 1 \\ 0 & \text{sinon} \end{cases}$$

- On a souvent des contraintes supplémentaires comme: $a_{ij} = 0$ si $j > i + \Delta$ ($\Delta = 2$ dans les modèles de Bakis)
- Les modèles gauche-droite permettent de modéliser des signaux qui évoluent avec le temps (ex: parole)

Détail d'implantation des HMMs

Scaling des $\alpha_t(i), \beta_t(i)$

$\alpha_t(i) = p(o_1 o_t, q_t = s_i|\lambda)$ nécessite le calcul de:

$$\left(\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(o_s) \right)$$

- Plus t est grand et plus le résultat de ce calcul tend vers 0. La précision des réels en C (ou équivalent) est insuffisante pour coder ces valeurs, plus t augmente (de l'ordre de 100).

$\Rightarrow$ Il faut normaliser $\alpha_t(i)$.

- **Idée:** on multiplie $\alpha_t(i)$ par une valeur qui ne dépend que de t et qui assure une bonne dynamique de $\alpha_t(i)$. On applique également le même coefficient à $\beta_t(i)$ (même problème de précision). À la fin, ces coefficients s'annulent.

Observations multiples

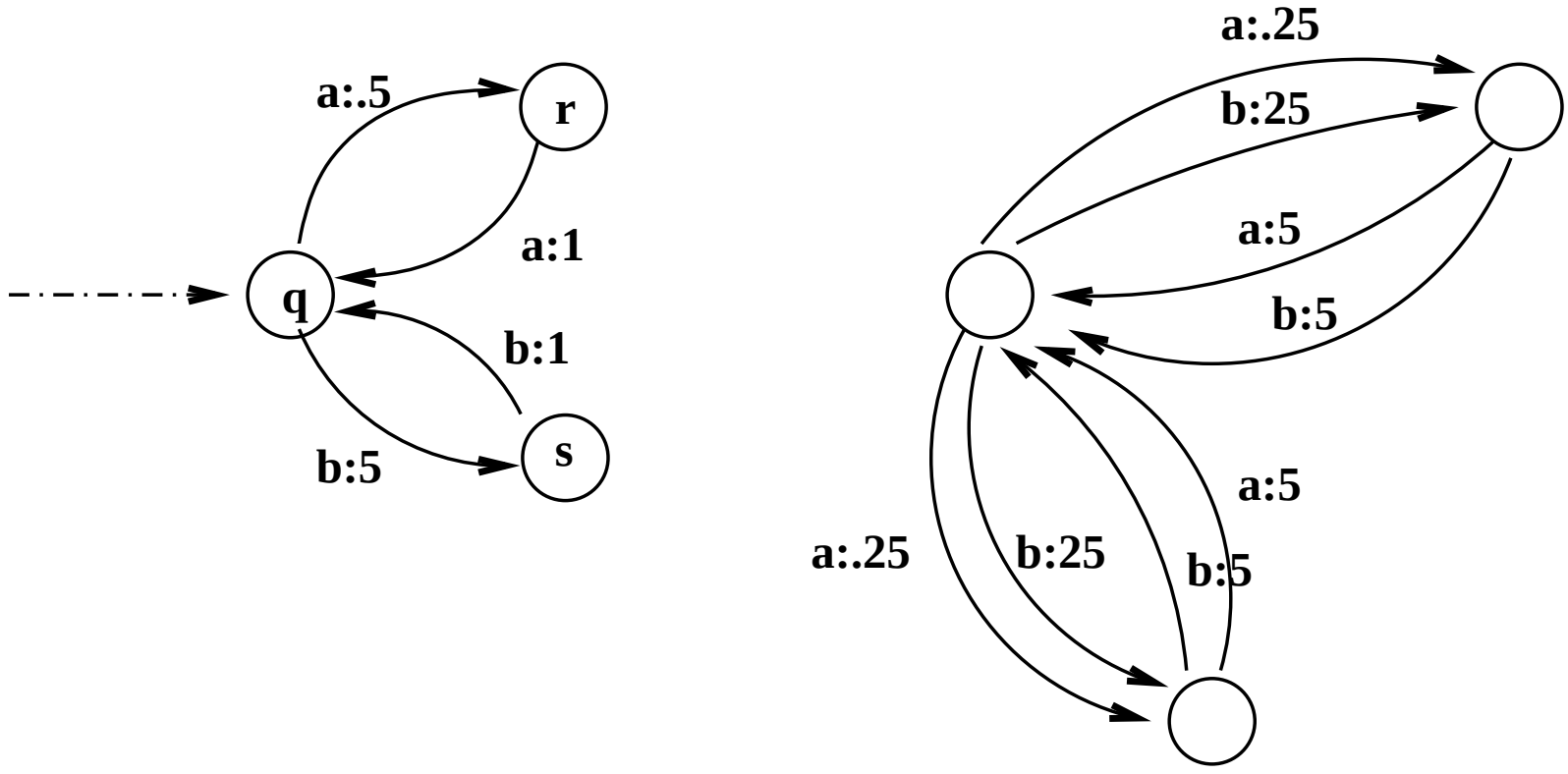
- Souvent, le corpus d'entraînement est présenté comme un ensemble de séquences d'observations (ex: phrases).
- Soit K le nombre de séquences:
  $O = [O^1, O^2, \dots O^K]$ avec $O^i = [O^i_1 O^i_2 \dots O^i_{T_i}] \ \forall i \in [1, K]$
- en supposant l'indépendance de chaque phrase:
  $p(O|\lambda') = \prod_{k=1}^{K} p(O^k|\lambda') = \prod_{k=1}^{K} P_k$
- Alors les formules de réestimations sont:

$$\overline{a_{ij}} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) a_{ij} b_j(o_{t+1}^k) \beta_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(j)}$$

$$\overline{b_j(l)} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1:o_t=v_l}^{T_k - 1} \alpha_t^k(i) \beta_t^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(j)}$$

Les points critiques

Exemple pris dans Charniak93



- Apprentissage sur le corpus $\{aabb\}$:
  - avec le second modèle, les estimées ne changeront pas
- ajouter du bruit lors de l'initialisation avant apprentissage

Problèmes d'EM

- Si une transition (par exemple) n'est pas utile pour la génération de O, alors cette transition va voir sa probabilité décroître à chaque itération.

  ▷ sur-entraînement

- EM converge vers un maximum local de la vraisemblance.

  ▷ importance des choix initiaux

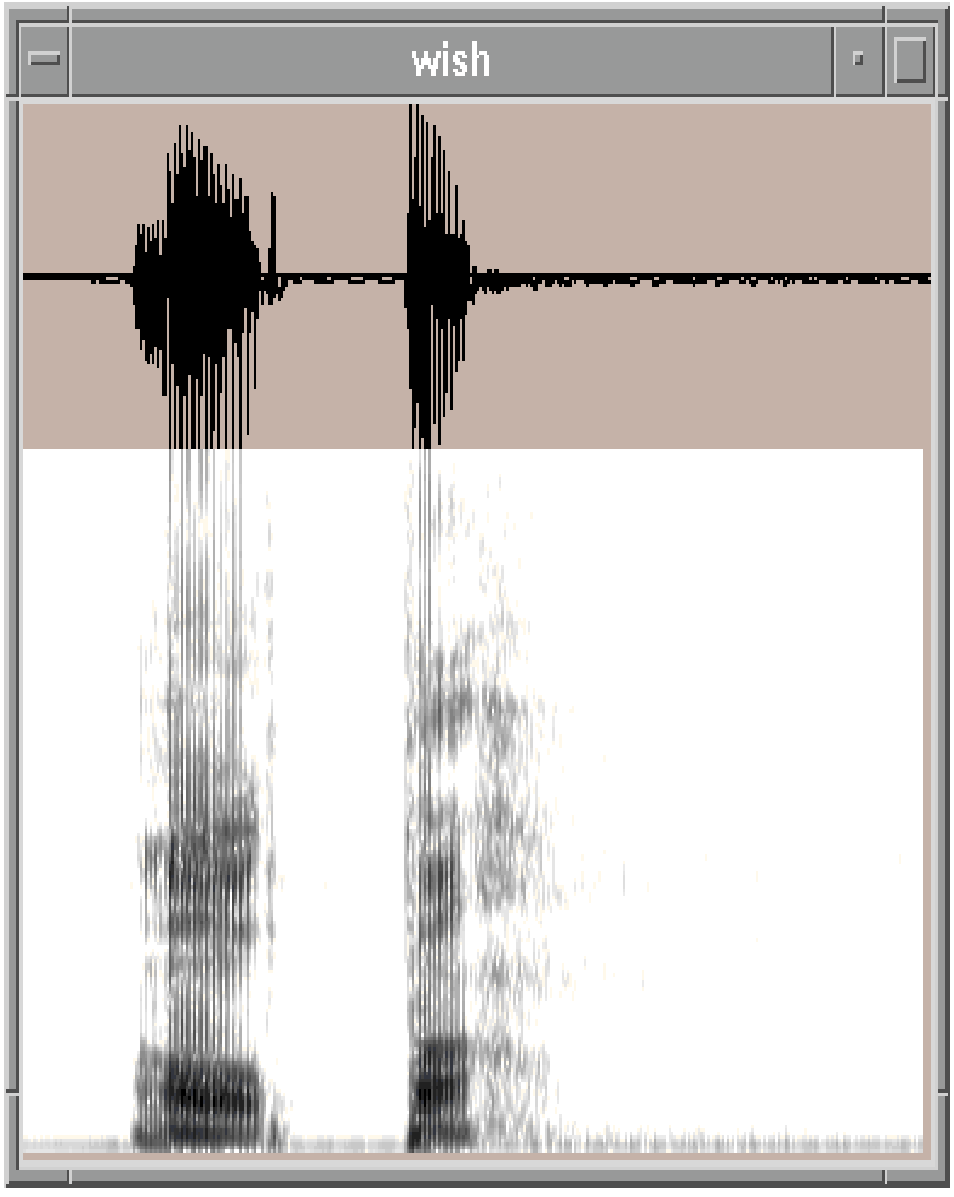
HMMs en reconnaissance de la parole (RAP)

À l'entrée du canal: une séquence de mots w
À la sortie du canal: un signal de parole O

$$\begin{aligned}
\hat{w} &= \text{argmax}_{w \in \mathscr{F}} \, p(w|O) \\
&= \text{argmax}_{w \in \mathscr{F}} \, \frac{p(O|w) \times p(w)}{p(O)} \\
&= \text{argmax}_{w \in \mathscr{F}} \, \underbrace{p(O|w)}_{\text{acoustico-phonétique}} \times \underbrace{p(w)}_{\text{modèle de langue}}
\end{aligned}$$

- Pour un bon panorama des techniques markoviennes utilisées dans les systèmes de RAP, lire (**Huang90**)

Représentation du signal

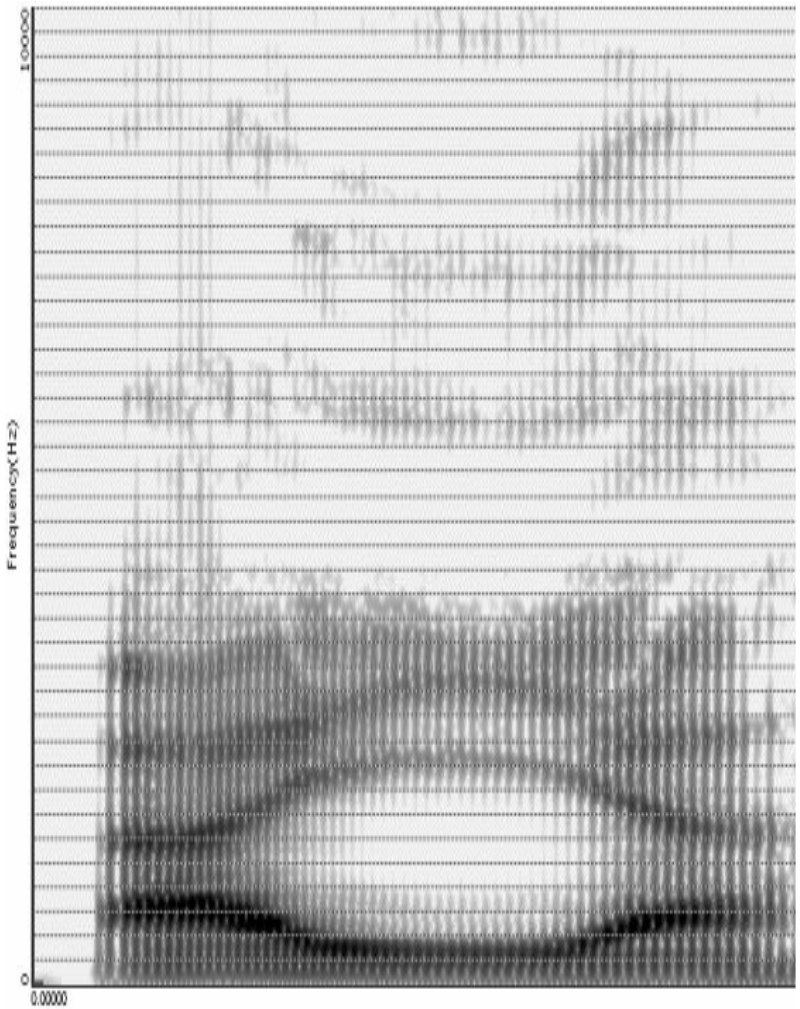


Typiquement:

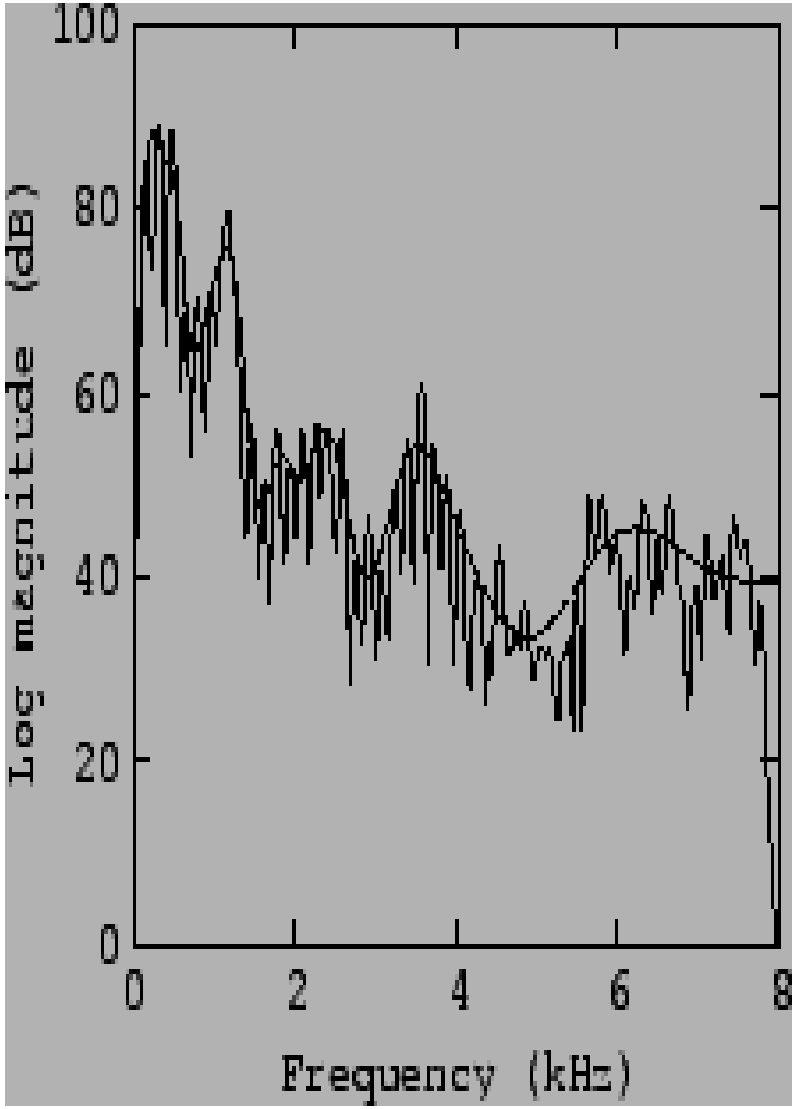
Signal 16 000 échantillons par seconde
(1 échantillon = 16 bits)

Spectro 39 coefficients par trame
(1 trame = 10 ms)
$\Longrightarrow$ 3 900 valeurs par seconde.

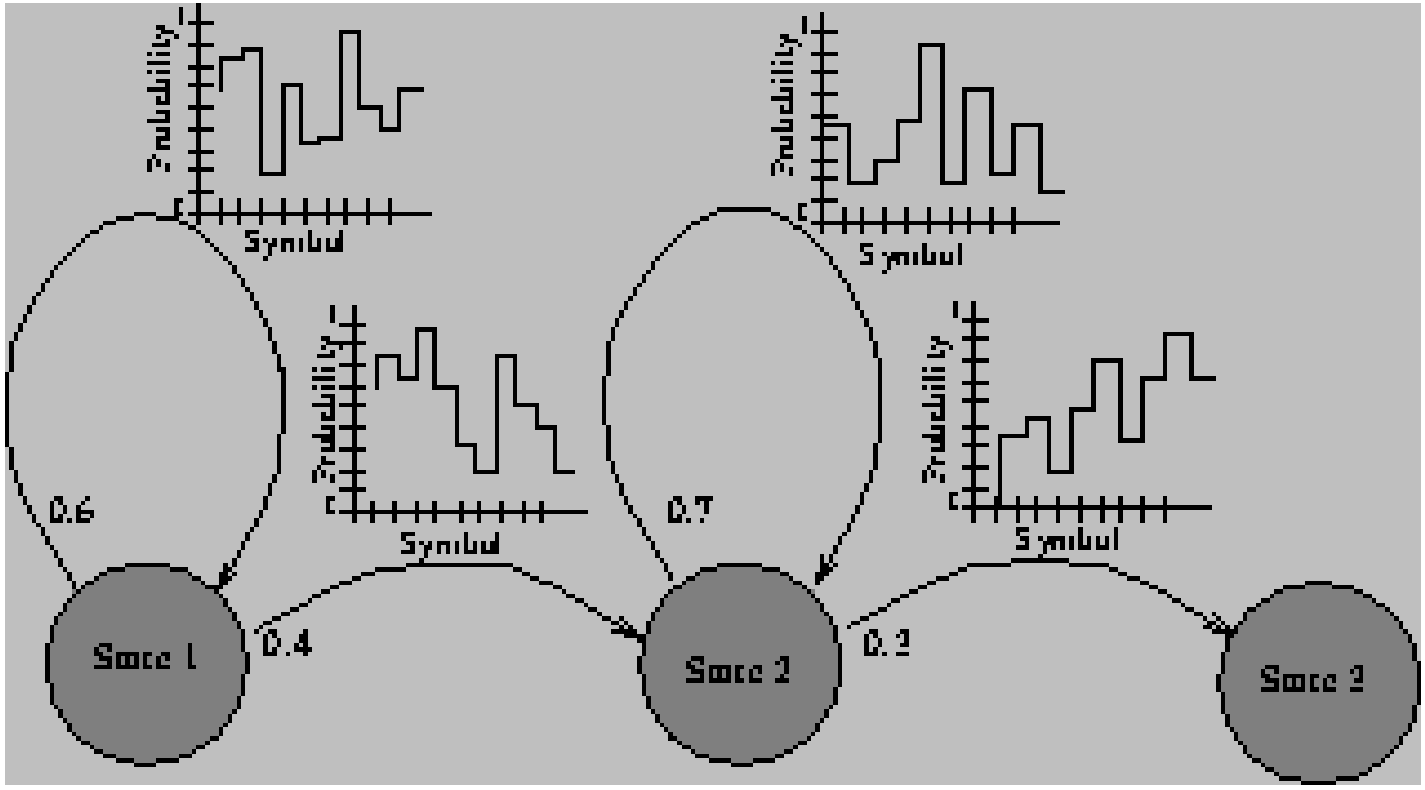
Représentation du signal



[a j a]

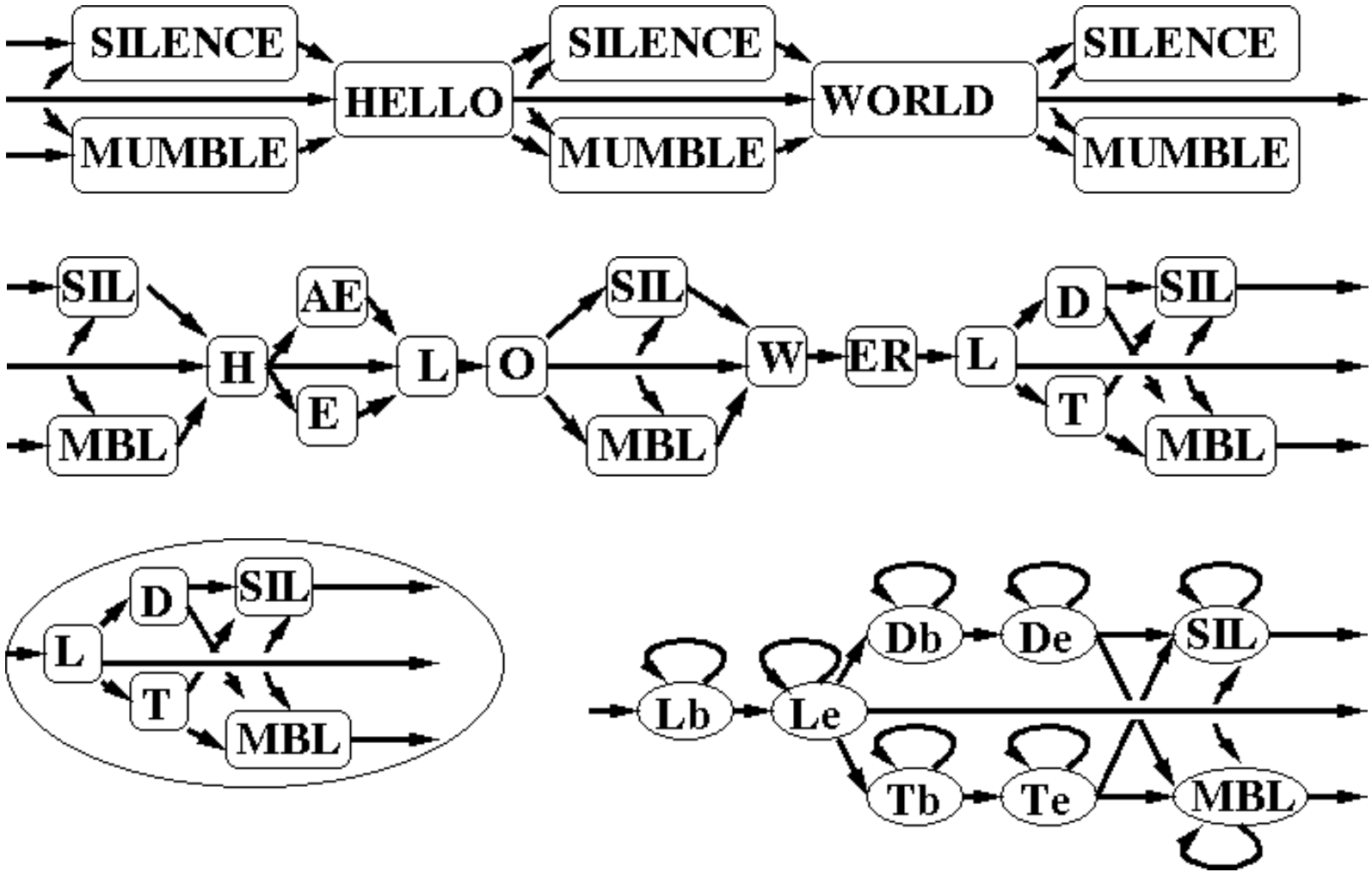


Exemple de modèle acoustique



Exemple de graphe de reconnaissance

http://isl.ira.uka.de/speechCourse/slides/



Références I