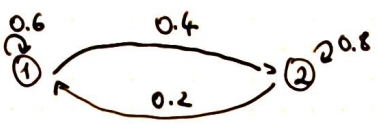
Graphes et Automates

Modèles de Markov cachés

Pratiquement, cela donne :

1. Avec $\lambda = (\pi, A, B)$ et $O = o_1, o_2, o_3, \dots, o_t$, il faut calculer $P(O|\lambda)$
2. Avec $\lambda = (\pi, A, B)$ et $O = o_1, o_2, o_3, \dots o_t$, il faut trouver $Q_i = \{q_1^i, q_2^i, \dots\}$ plus probable parmi les séquences d'états possibles $Q = \{Q_1, Q_2, Q_3, \dots\}$ $\max_{i \in Q} P(Q_i | O, \lambda)$
3. Avec $O = o_1, o_2, \dots, o_t$, trouver $\lambda = (\pi, A, B)$ qui maximise $P(O|\lambda)$

exemple



$B_1 \rightarrow \mathcal{N}(\mu_1, \sigma_1)$ dans $[0, +\infty[$ → vraisemblance

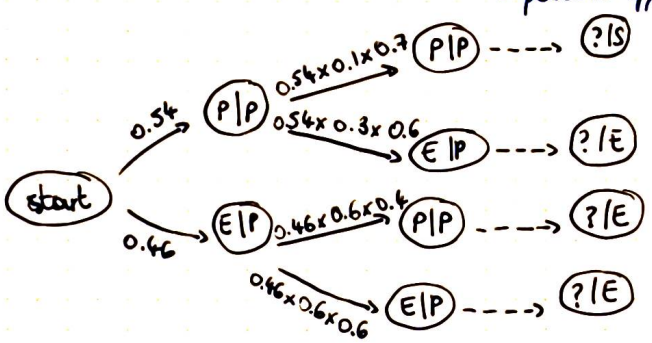
$B_2 \rightarrow \mathcal{N}(\mu_2, \sigma_2)$ ⚠ multiplication probas et vraisemblance

Algorithme non-polynomial - algorithme min-max

- tous les algos qui nécessitent de réaliser du back-tracking

Trouver la séquence la plus probable : générer toutes les solutions possibles grâce à un arbre de solutions

→ parcours effectué en profondeur



Algorithme de Viterbi : → simplifier l'arbre au fur et à mesure de sa construction

→ supprimer les branches de faible probabilité

→ parcours en largeur basé sur la programmation dynamique (création de sous-problèmes du plus petit au plus grand en stockant les résultats intermédiaires

→ utilise la propriété de l'homogénéité ($P(X_{n+1} = j \mid X_n = i)$)

Viterbi

* garder l'état le plus probable : à l'instant t, la séquence d'états la plus probable ne dépend que de la séquence d'états la plus probable à l'instant t-1.
* déduire la solution optimale d'un problème à partir d'une solution optimale d'un sous-problème.

Lire l'article en anglais sur la partie Forward/Backward et Viterbi