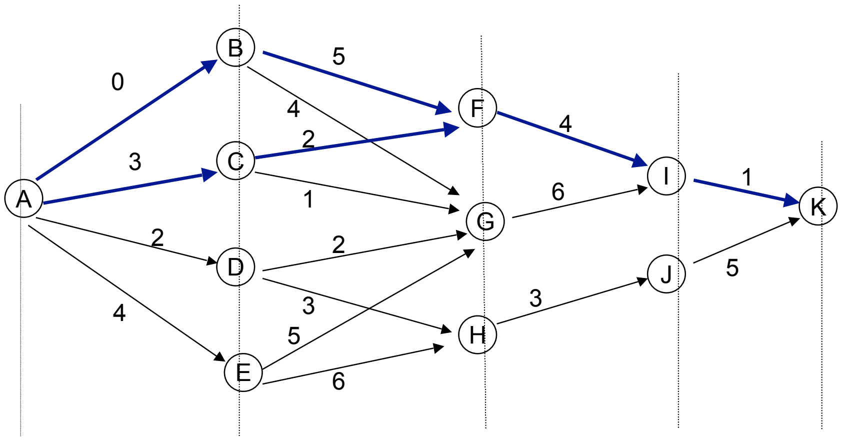
CNAM

Cours RCP208

PROGRAMMATION DYNAMIQUE

Fouad Badran

Exemple 1 : On veut relier une ville A à une ville Z par une voie ferrée. Le graphe ci-dessous présente les tronçons envisageable, valués par leur coût de construction



2 solutions de coûts minimum : (A,B,F,I,K) ou (A,C,F,I,K)

Processus de décision décomposable en phases

- Un processus qui se décompose en une suite de N phases de décision.
- Chaque phase k est caractérisée par un ensemble d'états initiales Y_{k-1} et un ensemble d'états terminales Y_k
- Au cours de la phase k, l'évolution d'un état initial y_{k-1} vers un état terminal y_{k-1} se réalise par l'intermédiaire d'une décision dont le coût est noté par $v_k(y_{k-1}, y_k)$.
- Une politique **d** est une succession de décisions $d_1 d_2 \dots d_N$, permettant de générer une séquence d'états : $y_0\, y_1 \dots y_{N-1}\, y_N$
- Le coût d'une telle politique est défini par :

$$\mathbf{c(d)} = \sum_{k=1}^{N} \mathbf{v_k(y_{k-1}, y_k)}$$

Deux problèmes se posent :

(P1) Etant donné un état initial y_0 et un état terminal y_N, déterminer la politique de coût optimal parmi celles qui évoluent de y_0 à y_N.

(P2) Déterminer une politique de coût optimal parmi celles qui évoluent d'un état donné initial vers l'un des états de Y_N

Principe de Bellman

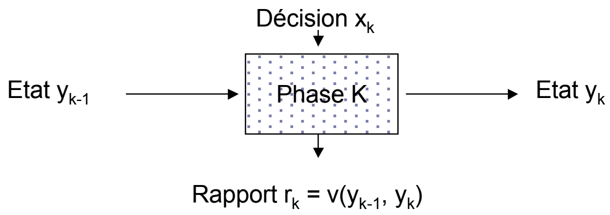
Définition (sous politique) : étant donné une politique **d**= $d_1 d_2 \dots d_N$ générant la séquence des états $y_0\, y_1 \dots y_{N-1}\, y_N$. La séquence de décisions $\mathbf{d^k}=d_1 d_2 \dots d_k$ qui génère la séquence $y_0\, y_1 \dots y_k$ sera dite une sous politique de **d**.

Principe de Bellman : « Toute sous politique d'un politique optimale est elle-même optimale ».

Ainsi si la politique **d**= $d_1 d_2 \dots d_k \dots d_N$, générant la séquence des états $y_0\, y_1 \dots y_k \dots y_{N-1}\, y_N$ est optimale parmi toutes celles qui permettent d'atteindre y_N à partir de y_0 alors la sous politique $\mathbf{d^k}= d_1 d_2 \dots d_k$ est optimale parmi toutes celles qui permettent d'atteindre y_k à partir de y_0.

$$\overbrace{\underbrace{y_0 \xrightarrow{d_1} y_1 \xrightarrow{d_2} y_2 \xrightarrow{d_3} \cdots\cdots \xrightarrow{d_k} y_k}_{\mathbf{d^k}} \xrightarrow{d_{k+1}} \cdots\cdots \xrightarrow{d_N} y_N}^{\mathbf{d}}$$

Formules d'optimisation séquentielles



- On note par $f_k(y_{k-1},y_k)$ le « coût » de la politique minimale permettant d'évoluer de y_0 à y_k (à la fin de l'étape k).
- On a $f_1(y_0,y_1) = v_1(y_0,y_1)$
- Le principe de Bellman nous permet de tirer la relation de récurrence :

$$f_k(y_0,y_k) = \underset{y_{k-1} \in D(y_0,y_k)}{MIN} \left[f_{k-1}(y_0,y_{k-1}) + v_k(y_{k-1},y_k) \right]$$

Où $D(y_0,y_k)$ est un sous ensemble de Y_{k-1} défini par :

$$y_{k-1} \in D(y_0,y_k)$$
$$\Leftrightarrow$$

y_{k-1} est l'avant dernier état d'une politique **d** permettant l'évolution de y_0 à y_k.

Exercice

Problème d'investissement : On suppose que l'on dispose de 4 unités monétaires (u.m.) que nous souhaitons investir dans 4 régions différentes.

D'autre part, on se fixe la règle que l'on ne peut pas investir plus de 3 u.m. dans une même région.

Le tableau ci-dessous indique l'apport qu'apporte l'investissement de x_i u.m. dans l'actions R_j.

xi \ R_j	R_1	R_2	R_3	R_4
x_1	0	0	0	0
x_2	25	24	18	17
x_3	38	39	36	37
x_4	47	53	54	55

Déterminer un plan d'investissement optimal.

Alignement de séquences

Le rôle de l'alignement des séquences est d'inférer l'évolution réelle entre deux séquences sans aucune connaissance à priori sur leur évolution.
Ainsi, considérons l'alphabet {A,C,G,T} et les deux séquences **a**=ACTGC et **b**=ACGTC, plusieurs alignements sont possibles, à titre d'exemple considérons les deux alignements suivants :

$$1\begin{cases}\text{ACTG -C}\\ \text{A -CGTC}\end{cases} \quad \text{et} \quad 2\begin{cases}\text{ACTG -C}\\ \text{AC -GTC}\end{cases}$$

Le premier alignement est de longueur L=6 et correspond à :

- La suppression du second caractère (C) de la séquence **a** dans la séquence **b** .
- La substitution du troisième caractère (T) de **a** par le second caractère (C) de **b**.
- L'insertion dans **b** du caractère T entre les deux caractères (G) et (C) de **b**.

Le second alignement est de longueur L=6 et correspond à :

- La suppression du troisième caractère (T) de la séquence **a** dans la séquence **b**.
- L'insertion dans **b** du caractère T entre les deux deux caractères (G) et (C) de **b**.

Longueur d'un alignement

Ex1

$\mathbf{a} = a_1\, a_2\, a_3\, a_4\, a_5\, a_6\, a_7\, a_8\, a_9\, a_{10}$

$\mathbf{b} = b_1\, b_2\, b_3\, b_4\, b_5\, b_6\, b_7\, b_8$

$$\left.\begin{array}{l}\mathbf{a^*} = -\ a_1\, a_2\, a_3\, a_4\, a_5\, a_6\, a_7\, a_8\, a_9\, a_{10}\\ \mathbf{b^*} = b_1\, b_2\, b_3\, -\ -\ b_4\ -\ b_5\, b_6\, b_7\, b_8\end{array}\right\}\ L=11$$

Ex2

$\mathbf{a} = a_1\, a_2\, a_3\, a_4\, a_5\, a_6\, a_7\, a_8\, a_9\, a_{10}$

$\mathbf{b} = b_1\, b_2\, b_3\, b_4\, b_5\, b_6\, b_7\, b_8$

$$\left.\begin{array}{l}\mathbf{a^*} = -\ -\ a_1\ -\ a_2\, a_3\, a_4\, a_5\, a_6\, a_7\, a_8\, a_9\, a_{10}\\ \mathbf{b^*} = b_1\, b_2\, b_3\, b_4\, b_5\ -\ -\ -\ -\ b_6\, b_7\, b_8\ -\end{array}\right\}\ L=13$$

Alignement suite

Ainsi, les trois opérations élémentaires dans un alignement sont :

1) $\begin{pmatrix}X\\Y\end{pmatrix}$ "Match" si $\begin{cases}X = Y \text{ identité}\\ X \neq Y \text{ substitution}\end{cases}$ 2) $\begin{pmatrix}X\\-\end{pmatrix}$ **Insertion** 3) $\begin{pmatrix}-\\Y\end{pmatrix}$ **Suppression**

Distance entre 2 séquences

Etant donné 2 séquences $\mathbf{a}=a_1a_2\ldots a_n$ et $\mathbf{b}=b_1b_2\ldots b_n$ écrites dans une alphabet fini ***A***, On suppose définit :

- d(a,b) = d(b,a) (coût de la substitution du caractère a par le caractère b ou b par a).
- d(a,a)=0 (coût du « match » identité)
- d(a,-) = d(-, a) = g(a) (coût de la substitution ou l'insertion de a).

Étant donné un alignement donné de **a** et **b** : $\mathbf{a^* = a_1^* a_2^* ... a_L^*}$ **et** $\mathbf{b^* = b_1^* b_2^* ... b_L^*}$

On définit le coût de cet alignement par $\sum_{i=1}^{L} d(a_i^*, b_i^*)$

La distance entre les deux mots **a** et **b,** D(**a**,**b**), est définie comme étant le coût de l'alignement minimal :

$$D(a,b) = \min_{(a^*,b^*)} \sum_{i=1}^{L} d(a_i^*, b_i^*)$$

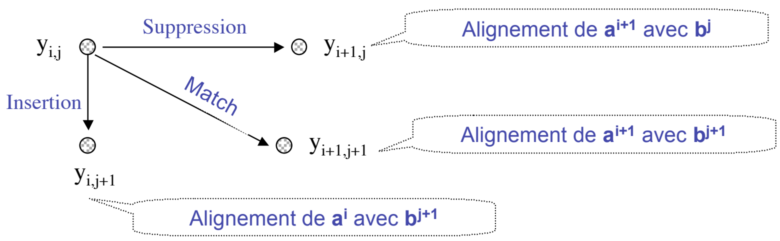
Calcule de la distance par Programmation dynamique

Étant donné 2 séquences $\mathbf{a}=a_1a_2\ldots a_n$ et $\mathbf{b}=b_1b_2\ldots b_n$, il est possible de définir un alignement des 2 séquences **a** et **b** par une séquence de décisions consécutives. On note par :

$\mathbf{a^i} = a_1a_2\ldots a_i$ la sous séquence de **a** formée des i premiers caractères

$\mathbf{b^j}=b_1b_2..b_j$ la sous séquence de **a** formée des i premiers caractères

Afin de définir le processus décisionnelle, on note par $y_{i,j}$ l'état qui représente un alignement particulier des 2 sous séquences $\mathbf{a^i}$ et $\mathbf{b^j}$. Étant donné l'état $y_{i,j}$ il est possible de progresser dans l'alignement en prenant l'une des 3 décisions suivantes :



Exemple

On considère les 2 séquences suivantes définies sur la même alphabet A = {A,C,G,T}:

a=GGGTGATTAG et **b**=GCGATATAG

Ainsi, l'alignement suivant :

a*= G – G G T G A T T – A – – G

b*= – G C – – G – G A T A T A G

1 Suppression

3 substitution

8 substitution

2 insertion

Correspond à la séquence de décisions suivantes : 1 suppression, 2 insertion, 3 substitution, 4 suppression, 5 suppression, 6 identique, 7 suppression, 8 substitution, 9 substitution, 10 insertion, 11 identique, 12 insertion, 13 insertion, 14 identique

Calcule de la distance

Les formules de la programmation dynamique permettent de calculer cette distance.

Soit **a**=$a_1a_2a_3\ldots a_n$ et **b**=$b_1b_2b_3\ldots b_m$

On définit $D_{i,j}=D(a_1a_2\ldots a_i,\ b_1b_2\ldots b_j)$

On a : $D_{00}= 0$, $\mathbf{D_{0,j}=\sum_{k=1}^{j} d(-,b_k)}$ et $\mathbf{D_{i,0}=\sum_{k=1}^{i} d(a_k,-)}$

Et la formule de récurrence : $\mathbf{D_{i,j} = MIN\left[D_{i-1,j} + d(a_i,-), D_{i-1,j-1} + d(a_i,b_j), D_{i,j-1} + d(-,b_j)\right]}$

$$\begin{tikzcd}
D_{i-1,j-1} \arrow[dr, "d(a_i{,}b_j)"'] & D_{i-1,j} \arrow[d, "d(a_i{,}-)"] \\
D_{i,j-1} \arrow[r, "d(-{,}b_j)"'] & D_{ij}
\end{tikzcd}$$

Exemple de calcul

a=GGGTGATTAG et **b**=GCGATATAG

	-	G	C	G	A	T	A	T	A	G
-	0	1	2	3	4	5	6	7	8	9
G	1	0	1	2	3	4	5	6	7	8
G	2	1	2	1	2	3	4	5	6	7
G	3	2	3	2	3	4	5	6	7	6
T	4	3	4	3	4	3	4	5	6	7
G	5	4	5	4	5	4	5	6	7	6
A	6	5	6	5	4	5	4	5	6	7
T	7	6	7	6	5	4	5	4	5	6
T	8	7	8	7	6	5	6	5	6	7
A	9	8	9	8	7	6	5	6	5	6
G	10	9	10	9	8	7	6	7	6	5

D(a,b)=5