REVUE FRANÇAISE D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

A. DELEDICQ

Programmation dynamique discrète. *k*-optimums d'un problème séquentiel

Revue française d'informatique et de recherche opérationnelle, tome 2, n° V2 (1968), p. 13-32

<http://www.numdam.org/item?id=RO_1968__2_2_13_0>

R.I.R.O.
(2e année, N° 11, 1968, p. 13-32)

PROGRAMMATION DYNAMIQUE DISCRETE

k-optimums d'un problème séquentiel.

par A. Deledicq (1)

Résumé. — *Cet article expose un nouvel algorithme de détermination des stratégies (ou politiques dans le cas déterministe) k-optimales pour un processus représentable par un graphe séquentiel.*

Le paragraphe 1 précise les notations. Les conséquences de la notion k-optimalité sont exposées au paragraphe 2. Les exemples qui ont servi à tester l'algorithme présenté dans le paragraphe 3, sont décrits dans le paragraphe 4. Enfin dans le paragraphe 5 figurent les résultats montrant que l'exploration du voisinage de l'optimum coûte très peu lorsqu'on utilise l'algorithme à réflexion.

1. GRAPHES SEQUENTIELS. NOTATIONS

Parmi les problèmes dont la résolution relève de la programmation dynamique discrète, il en est qui sont représentables par un graphe séquentiel. Nous dirons que de tels problèmes sont séquentiels [5].

On peut remarquer que pour qu'un problème d'optimisation soit séquentiel, il est nécessaire que la fonction à optimiser puisse être mise sous la forme d'une somme finie de termes, chaque terme étant fonction d'une variable de décision, pouvant prendre un nombre fini de valeurs discrètes.

Pour des programmes dynamiques discrets non séquentiels, au sens ainsi défini, (exemple : problème à horizon infini), le tracé d'un graphe représentant partiellement le problème, aide souvent à la compréhension.

Les notations utilisées sont très proches de celles de A. Kaufmann et R. Cruon [1] [2] [5].

(1) Commissariat à l'Énergie Atomique (DPg-SEEG), Faculté des Sciences de Paris.

A. Cas déterministe

Soit un graphe séquentiel (G, Γ) (fig. 1).

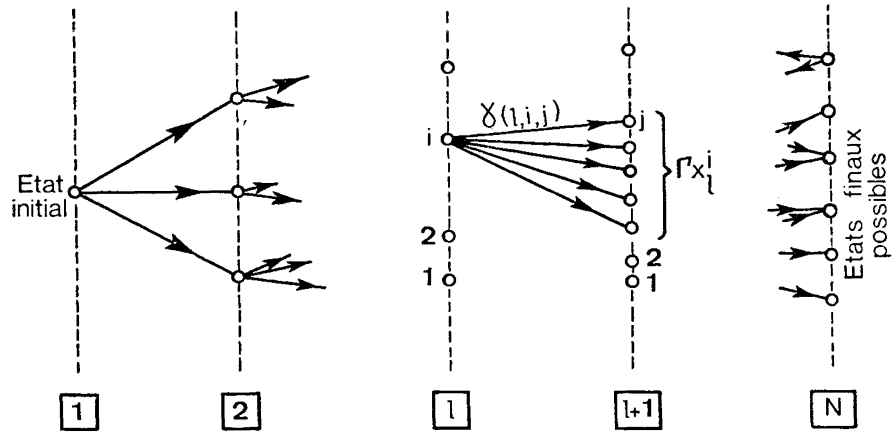

Figure 1

Nous dirons que 1, 2, — l, N, sont les « instants » où l'on doit prendre une décision.

A l'instant l l'état du processus est repéré par un sommet x_l^i.

Une décision prise à l'instant l consiste à choisir dans Γx_l^i un sommet x_{l+1}^j. Le coût de cette décision est $\gamma(l, i, j)$.

Une politique à partir d'un sommet x_l^i est une suite de sommets

$$\{ x_l^i, x_{l+1}^{j(l+1)}, x_{l+2}^{j(l+2)}, \ldots x_n^{j(n)} \}$$

repérant les états successifs du processus depuis l'instant l jusqu'à l'instant final N

$$x_{l+m+1}^{j(l+m+1)} \in \Gamma x_{l+m}^{j(l+m)}$$

Nous supposons l'état initial x_1^1 fixé ; s'il ne l'était pas, il suffirait d'introduire un état x_0, relié à tous les états x_1^i par des arcs de coût nul, et de changer le numérotage.

L'objectif est de déterminer une politique à partir de x_1^1, minimisant le coût total C :

$$C = \sum_{l=1}^{N-1} \gamma(l, i(l), j(l))$$

avec :

$$i(1) = 1$$
$$i(l) = j(l-1) \qquad \text{pour} \qquad l = 2, \ldots, N$$
$$j(l) \in \Gamma x_l^{i(l)} \qquad \text{pour} \qquad l = 1, \ldots, N$$

et d'obtenir les politiques k-optimales.

B. Cas aléatoire

Après chaque décision, intervient une phase de hasard. Le processus est alors appelé processus D — H. Les sommets précédant une phase de hasard sont notés y_l^i (fig. 2).

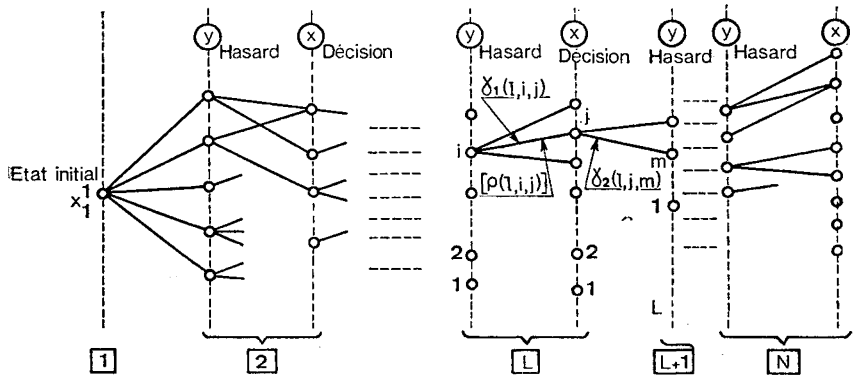

Figure 2

Ainsi

$$\bigcup_j y_l^j = \bigcup_i \Gamma x_{l-1}^i, \quad \bigcup_j x_l^j = \bigcup_i \Gamma y_l^i$$

Une décision prise à l'instant l consiste à choisir dans Γx_l^i un sommet y_{l+1}^j. Le coût de cette décision est $\gamma_2(l, i, j)$.

D'un sommet y_l^i le hasard peut amener à un sommet $x_l^j \in \Gamma y_l^i$. Le coût de cette éventualité est $\gamma_1(l, i, j)$, sa probabilité est $p(l, i, j)$.

Soit Λ une application telle que :

$$\begin{cases} \Lambda x_l^i \in \Gamma x_l^i & l = 1, 2, \ldots, N-1 \\ \Lambda y_l^j = \Gamma y_l^j & l = 2, 3, \ldots, N \end{cases}$$

une stratégie à partir d'un sommet x_l^i (resp. y_l^j) est une arborescence (A_l^i, Λ) (resp. (B_l^i, Λ)) de centre x_l^i (resp. y_l^j).

A_l^i(resp. B_l^i) est l'ensemble des sommets de l'arborescence ; les sommets de A_l^i sont donc pris parmi ceux de G, un même sommet de G pouvant apparaître plusieurs fois dans A_l^i (il convient alors de distinguer ces diverses occurrences).

Une stratégie consiste donc à choisir les décisions successives à prendre, pour toutes les éventualités pouvant apparaître.

L'objectif est de déterminer une stratégie à partir de x_1^1 minimisant l'espérance mathématique du coût total :

$$C = E(A_1^1, \Lambda)$$

avec :

$$E(A_l^i, \Lambda) = \gamma_2(l, i, j) + E(B_l^j, \Lambda)$$
$$E(B_l^j, \Lambda) = \sum_m p(l, j, m)[E(A_{l+1}^m, \Lambda) + \gamma_1(l, j, m)]$$
$$E(A_N^i, \Lambda) = 0$$

et les stratégies k-optimales.

2. k-OPTIMALITE

Dans un problème discret, les politiques (ou stratégies) peuvent être ordonnées suivant leurs coûts associés C croissants : politique 1-optimale, 2-optimale, ...

Une politique P_1 (resp. stratégie S_1) est dite incluse dans une politique P_0 (resp. S_0) si la suite (resp. l'arborescence) représentant P_1 (resp. S_1) est incluse dans celle représentant P_0 (resp. P_1). P_1 (resp. S_1) est une sous-politique (resp. stratégie) de P_0 (resp. P_1).

Théorème de k-optimalité [3] [1] [2]

Toute sous-politique (resp. stratégie) d'une politique (resp. stratégie) k-optimale est i-optimale avec $i \leqslant k$.

Ce théorème contient le principe d'optimalité ($k = 1$).

Intérêt de la k-optimalité

a) Soit à « étudier » un processus (économique): $\mathcal{E}$.

Le modèle E, que nous choisissons pour représenter $\mathcal{E}$, en est une schématisation, qui laisse de côté :

— la structure fine de $\mathcal{E}$ (« les détails »),

— les composantes difficilement formalisables de $\mathcal{E}$ (humaines, psychologiques, ...),

— ce que « nous ignorons plus ou moins » de $\mathcal{E}$ (fig. 3).

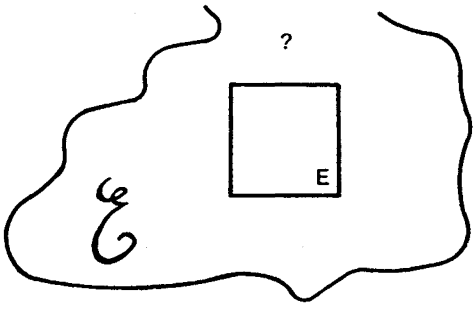

Figure 3

Il est donc bien évident que la politique optimale donnée par E, n'est pas forcément « la meilleure », pour $\mathcal{E}$. Il y a donc intérêt à dégager des politiques qui, relativement à E, sont « presque aussi bonnes » que la politique optimale.

Lorsque E est un graphe séquentiel, *l'exploration du voisinage* consiste à rechercher les politiques k-optimales.

b) On peut traiter le problème E en *supprimant certaines contraintes* (c'est-à-dire en rajoutant des arcs ou graphe) et chercher quel est alors l'ordre d'optimalité de la politique optimale antérieure. On mesure ainsi le degré de restriction qu'apporte la contrainte.

c) La détermination des politiques k-optimales permet d'atteindre des *optimums relatifs*, ainsi que le montre la figure 4.

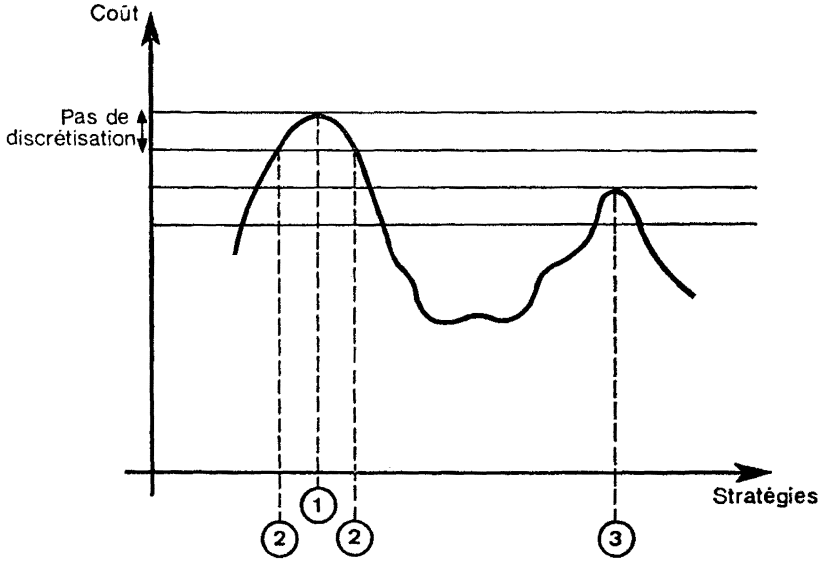

Figure 4

On voit sur cet exemple que dans le cas de problème discret par artifice (1), on ne doit pas choisir un pas de discrétisation h trop faible car l'ordre d'optimalité de l'optimum relatif augmente lorsque h diminue (évidemment h est, par contre, limité supérieurement par la précision désirée). Des calculs inutiles sont alors effectués car les premières politiques sous-optimales trouvées sont des *politiques parasites*. (Cette remarque se nuance lorsque l'optimum est situé sur la frontière du domaine des variables d'état.)

Dans le cas de problèmes aléatoires ce phénomène est important. En

(1) Voir § 4.

effet deux arborescences peuvent différer de quelques sommets sans que les structures globales des stratégies présentent une différence significative.

EXEMPLE (fig. 5) :

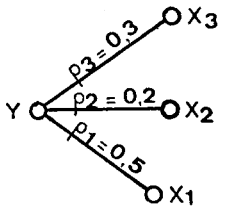

Figure 5

Supposons que la stratégie 2-optimale en y consiste à suivre en :

x_1 la politique 1-optimale
x_2 la politique 2-optimale
x_3 la politique 1-optimale

et que la stratégie 3-optimale en y consiste à suivre en :

x_1 la politique 1-optimale
x_2 la politique 1-optimale
x_3 la politique 2-optimale

On voit que ces stratégies ne diffèrent pas fondamentalement et l'on imagine le nombre de stratégies parasites lorsque le même phénomène se produit en tous les sommets de hasard y surtout si $|\Gamma y|$ est élevé.

La détermination des k-optimums ne paraît intéressante que lorsque la réponse au hasard doit être rapide.

La figure 6 résume ces considérations :

k-optimalité	Déterministe (D)	Aléatoire (DH)	
Discret par essence (E)	Intérêt fondamental	Intérêt limité	
Discret par artifice (A)	Apparition de politiques parasites	Stratégies parasites. Intérêt très limité	Problème du pas de discrétisation
		Stabilité de la structure des stratégies	

Figure 6

3. ALGORITHME A REFLEXION

Pour rechercher les chemins k-optimaux dans un graphe déterministe ou aléatoire, Kaufmann et Cruon [1] [2] ont proposé un algorithme plus rapide que celui initialement exposé par Bellman et Kalaba.

Cet algorithme (alg K.C.) est fondé sur la remarque suivante :

Supposons trouvée la politique 1-optimale à partir de x et qu'elle consiste à choisir d'abord $y \in \Gamma(x)$ et à suivre à partir de ce sommet la politique 1-optimale.

Alors la politique 2-optimale à partir de x est la meilleure des politiques suivantes :

— choisir y et suivre la politique 2-optimale à partir de ce sommet ;

— choisir $y' \in \Gamma(x)$, $y' \neq y$, et suivre la politique 1-optimale à partir de ce sommet.

Pratiquement, la méthode consiste à associer à chaque arc (x, y) un indice α, variable dans le cours du calcul, qui indique (si l'on calcule la politique i-optimale) que α des politiques μ-optimales à partir de $x (\mu = 1, 2, \ldots i - 1)$ passent par cet arc.

Le calcul s'effectue des x_{N-1}^j vers x_1^i, en déterminant en tous les sommets du graphe, les politiques 1, 2, ... k-optimales à partir de chacun d'entre eux.

Nous présentons ici un nouvel algorithme, que nous dirons « à réflexion » (alg R). Nous avons testé alg (K.C.) et agl (R) sur les problèmes exposés au paragraphe 4. Il apparaît que al g(R) est beaucoup plus rapide que alg (K.C.).

Il est fondé sur la remarque suivante :

Supposons trouvée la politique 1-optimale à partir de x_0.

Alors, pour connaître la politique 2-optimale à partir de x_0, il suffit de connaître les sous-politiques 2-optimales à partir des sommets du graphe appartenant à la politique 1-optimale, et les sous-politiques 1-optimale à partir des autres sommets, et celles-là seulement.

De même pour connaître la politique k-optimale à partir de x_0, il est inutile de connaître toutes les politiques i-optimales ($i \leqslant K$) à partir de tous les sommets du graphe.

Nous exposerons ces algorithmes dans le cas déterministe et nous donnerons les organigrammes du calcul dans le cas déterministe et dans le cas aléatoire.

Formules de calcul :

Le coût de la politique k-optimale à partir du sommet x_l^i est noté $f^k(l, i)$

A un instant donné du calcul :

$\sigma(l, i) = k$ signifie que l'on connaît les politiques 1-optimales, 2-optimales, ... k-optimales à partir du sommet x_l^i.

$\alpha(l, i, j) = q$ signifie que parmi ces k politiques, q consistent à choisir le sommet x_{l+1}^j.

* Alg (K.C.) n'utilise que l'indice α. On a :

$$f^1(N, i) = 0 \qquad \forall_i$$

$$f^k(N, i) = +\infty \qquad \forall_i \; ; k = 2, 3, \ldots K$$

$$\alpha(l, i, j) = 0 \qquad \forall_{i,j} \; ; l = 1, 2, \ldots N - 1$$

Successivement pour $l = N - 1, N - 2, \ldots 1$; pour chaque x_l^i et pour $k = 1, 2, \ldots K$, on calcule :

$$f^k(l, i) = \min_{x_{l+1}^j \in \Gamma x_l^i} [\gamma(l, i, j) + f^{\alpha(l,i,j)+1}(l + 1, j)]$$

Si $\hat{j}$ réalise le minimum, on augmente alors $\alpha(l, i, \hat{j})$ d'une unité.

* Alg (R) utilise les deux indices α et σ mais ne calcule que ce qui est strictement nécessaire pour déterminer les politiques 1, 2, ... K-optimales à partir de x_1^1.

En effet on peut calculer $f^k(l, i)$ dès que $\sigma(l + 1, j) \geqslant \alpha(l, i, j)$ pour tout j tel que $x_{l+1}^j \in \Gamma x_l^i$.

Pour chaque k, les calculs qu'il est nécessaire d'effectuer sont déterminés à partir de x_1^1 dans une première phase ; les calculs sont effectués grâce aux formules dans une deuxième phase.

Ce double balayage nécessite l'utilisation d'une « pile de mémoire » contenant les sommets rencontrés dans la première phase.

Remarquons qu'il est nécessaire de calculer la politique 1-optimale à partir de tous les sommets du graphe, on peut donc la calculer avant d'utiliser l'algorithme.

Les figures 7 et 8 donnent une idée du déroulement des calculs :

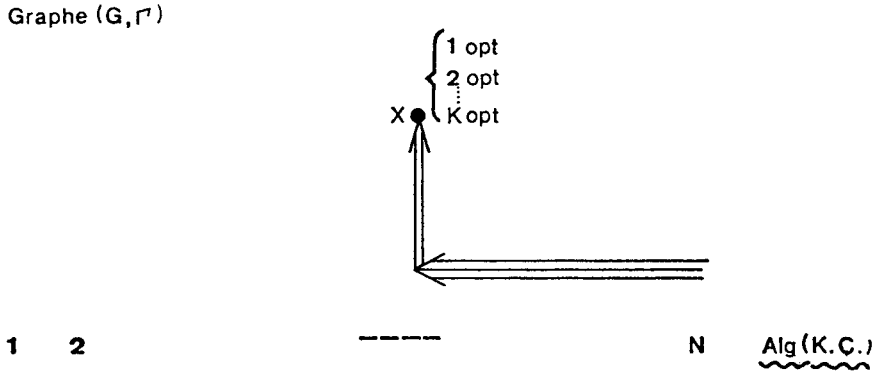

Figure 7

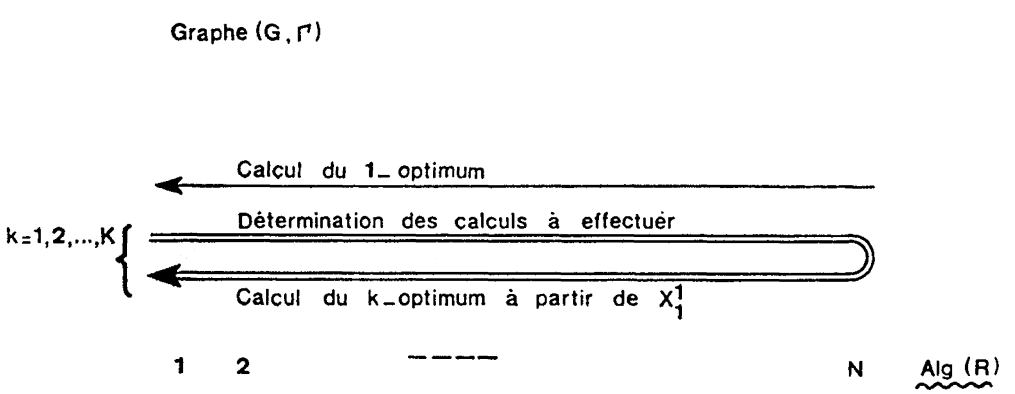

Figure 8

ORGANIGRAMME ALG(R) DETERMINISTE

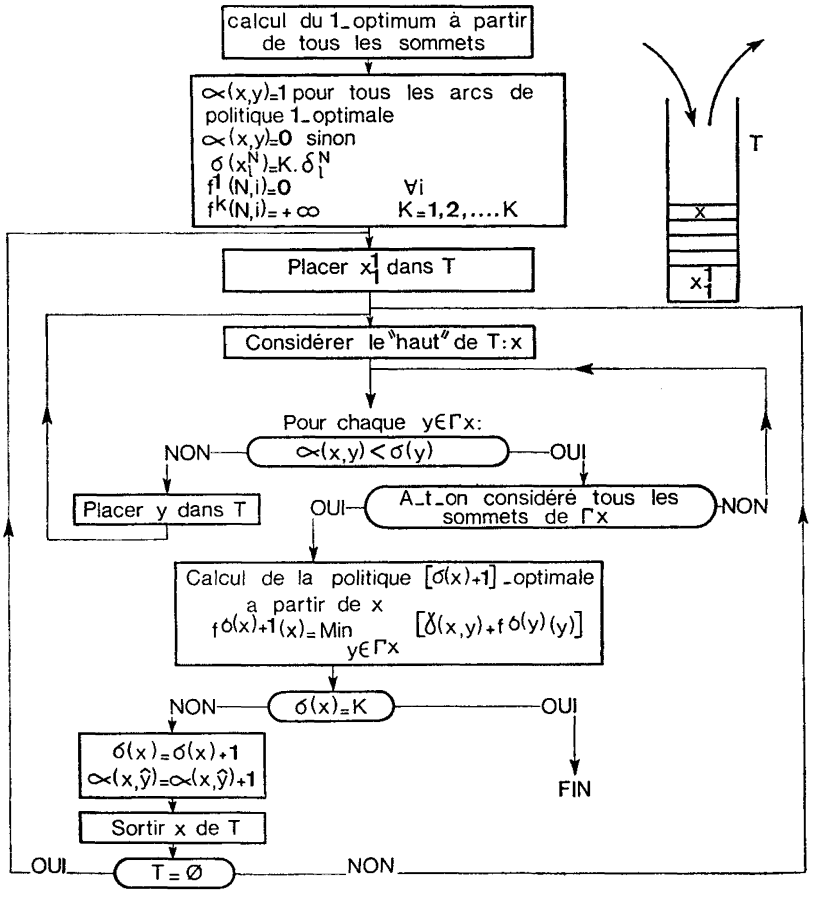

Figure 9

Nous donnons (fig. 9 à 12) les organigrammes de calcul pour alg (R) dans les cas déterministes et aléatoires. Nous avons préféré y être quelque peu « littéraire » plutôt que classiquement ésotérique. Ainsi nous avons évité les ordres de lecture et d'impression, l'emploi de variables intermédiaires et les divers raffinements de programmation, tels la gestion de la pile T, l'utilisation de coût infini ou de probabilité nulle pour des arcs fictifs... x et y représentent des sommets, on utilise ainsi un indice de moins dans les formules des organigrammes.

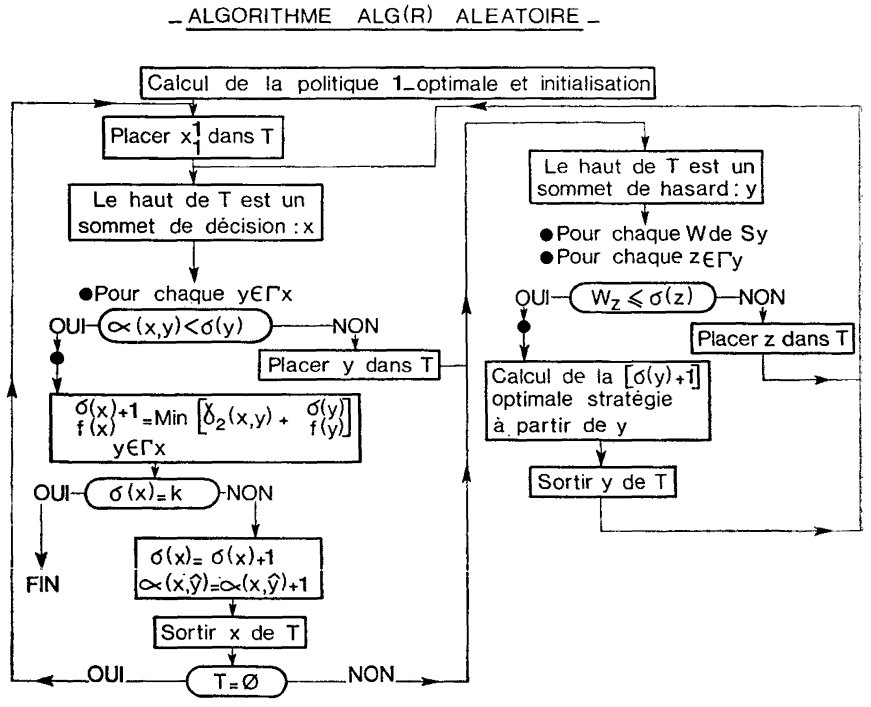

Figure 10

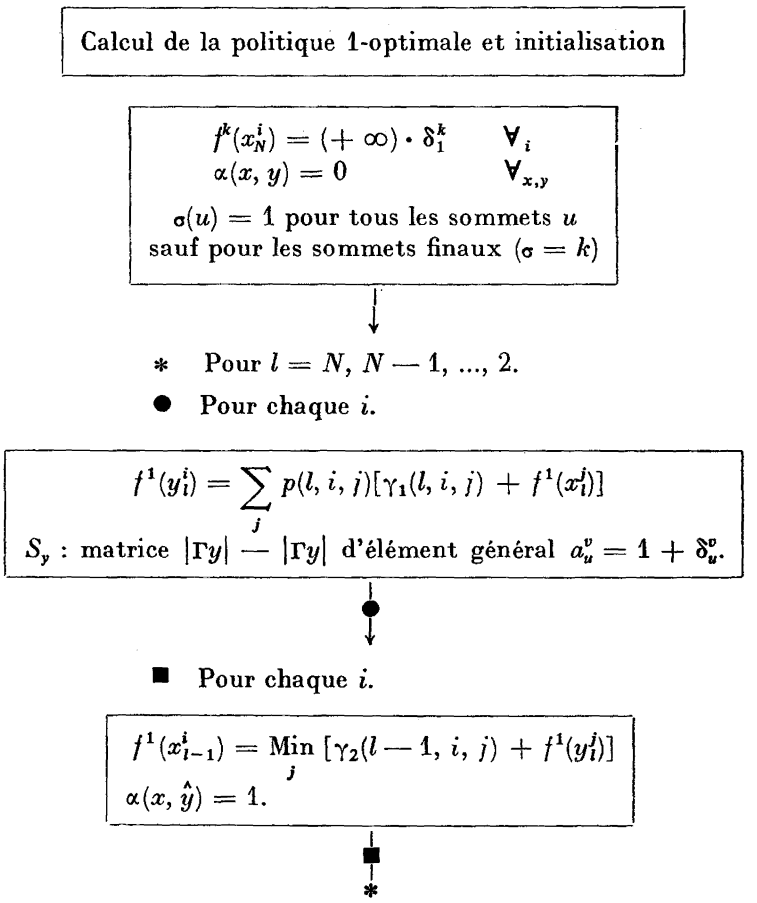

Figure 11

Pour le cas aléatoire, on utilise l'indice σ pour tous les sommets mais l'indice α n'est utilisé que pour les arcs issus d'un sommet de décision. La procédure de calcul est donc, pour les sommets de décision x, analogue au cas déterministe. Par contre pour les sommets de hasard y, on consultera les organigrammes partiels décrivant le déroulement des opérations. Une stratégie en y est en effet caractérisée par l'ordre d'optimalité de la stratégie à suivre en chaque $x \in \Gamma y$. On a introduit le tableau des stratégies candidates à la $[\sigma(y) + 1]$-optimalité à partir de y : S_y. Ses éléments sont des vecteurs W à $|\Gamma_y|$ composantes. Si la $m^{\text{ième}}$ composante de W vaut w_m, cela signifie que la stratégie considérée en y consiste à suivre la stratégie w_m-optimale à partir de x^m, si le hasard amène à $x^m \in \Gamma y$ (dans l'organigramme x^m est noté z et w_m est noté w_z afin de simplifier l'écriture).

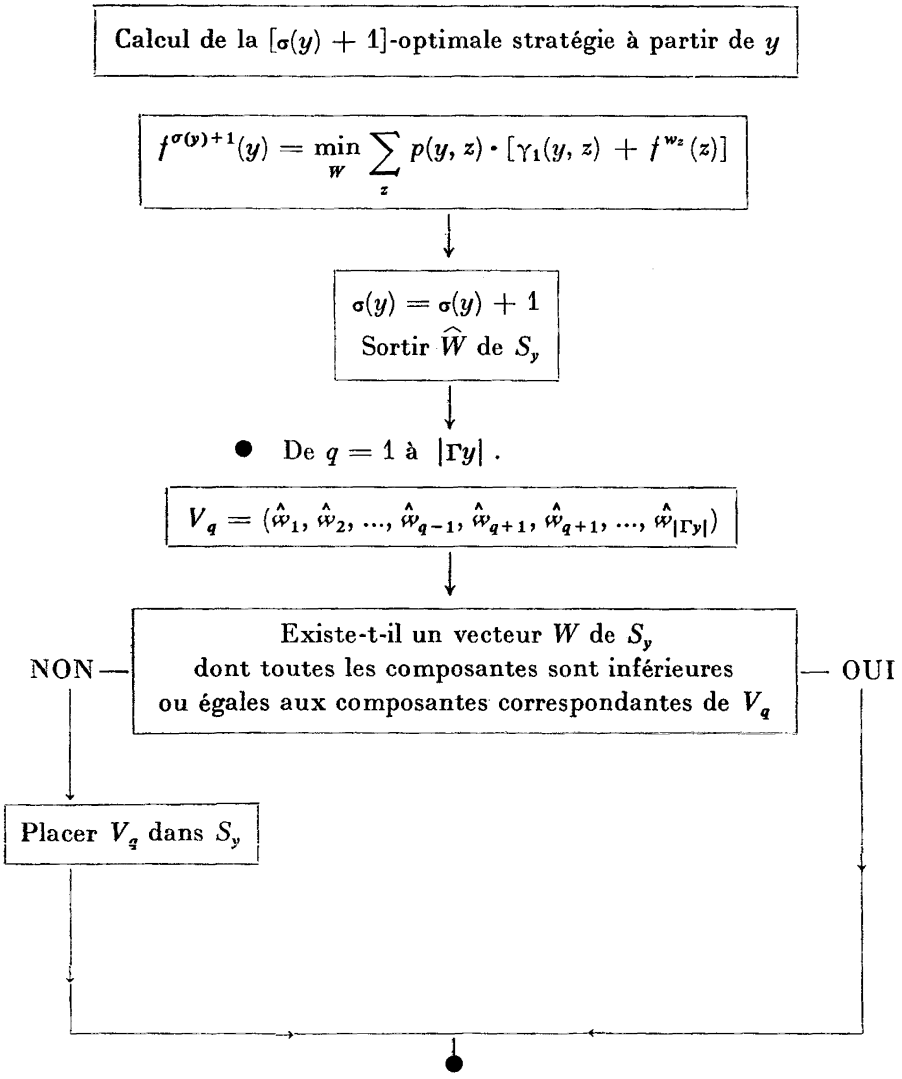

Figure 12

4. UNE CLASSIFICATION DES PROGRAMMES DYNAMIQUES DISCRETS

Outre la distinction entre programme déterministe et programme aléatoire, on peut classer les problèmes séquentiels en 2 catégories :

— les problèmes *discrets par essence*,

— les problèmes *discrets par artifice :*

Les variables d'états ou de décision varient continûment, mais on a discrétisé le problème en choisissant un certain pas de variation.

Le tableau de la figure 13 résume la classification et indique des exemples :

Programmes séquentiels	Déterministe	Aléatoire
Discret par essence	Choix entre N possibilités pour satisfaire une production donnée (problème D.E.)	Remplacement de matériel (problème D.H.E.)
Discret par artifice	Problème de distorsion de débit (problème D.A.)	Gestion de stock (problème D.H.A.)

Figure 13

Exemple de problème (D.E.)

Produire P objets à l'aide de 3 machines, le coût de production de p_i objets par la machine i étant $f_i(p_i)$.

Fonction à minimiser : $\sum_i f_i(p_i)$

Contrainte : $\sum_i p_i = p$

Graphe (fig. 14) :

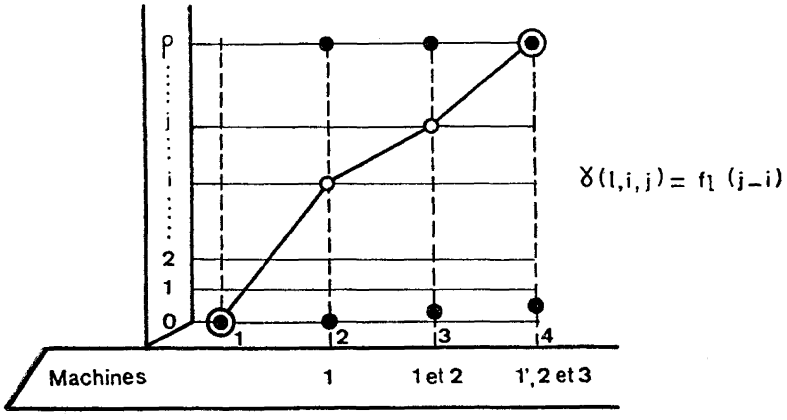

Figure 14

Exemple de problème (D.A.)

Étant donné un fluide radio-actif circulant dans 7 tubes (voir fig. 15), à débit total constant, déterminer la dose maximale en un point donné lorsqu'on suppose que les débits dans chaque tube peuvent s'écarter de 30 % de leur valeur moyenne.

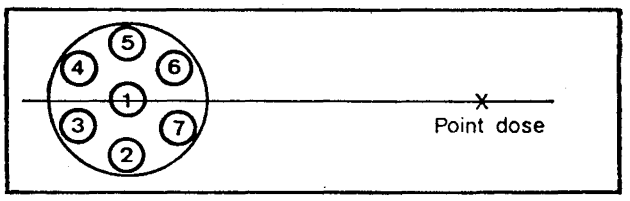

Figure 15

Fonction à minimiser : $\sum_{i=1}^{7} \alpha_i x_i (1 - e^{\frac{-k}{x_i}})$ k, α_i connus

Contraintes : $\Sigma x_i = 7$

$0{,}7 \leqslant x_i \leqslant 1{,}3$

Graphe : figure 16.

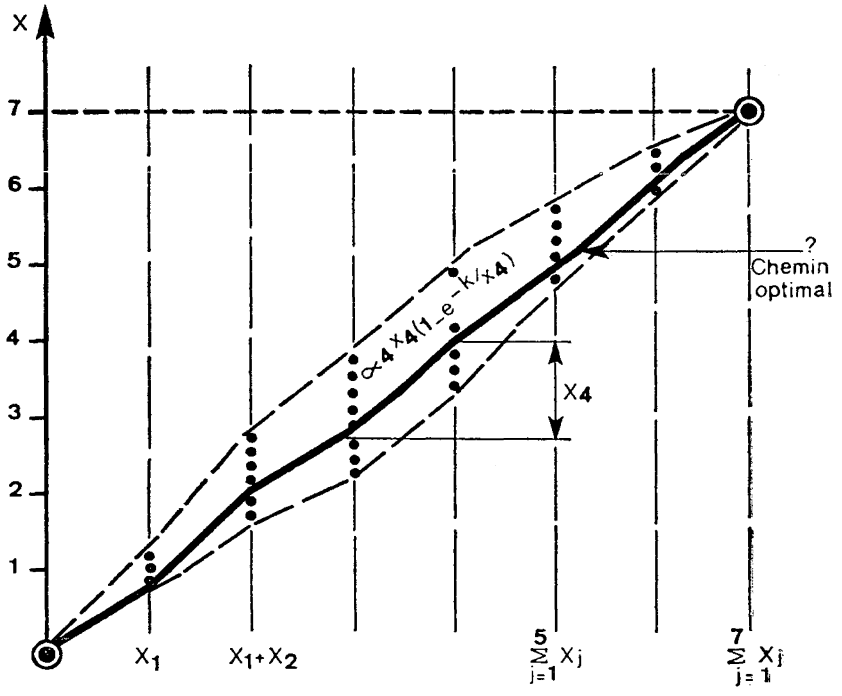

Figure 16

Exemple de problème (D.H.E.)

Soit à gérer pendant 20 ans un matériel de coût d'entretien annuel Va^2. La n-ième année le coût d'entretien est $Va^2/(1+\tau)^n$. Lorsqu'il atteint l'âge a, sa probabilité de défaillance est $a/10$ (1 si $a > 10$) et cela coûte $\gamma_1(a)/(1+\tau)^n$; son coût de remplacement est $\gamma_2(a)/(1+\tau)^n$. Le graphe de la figure 17 peut représenter le problème :

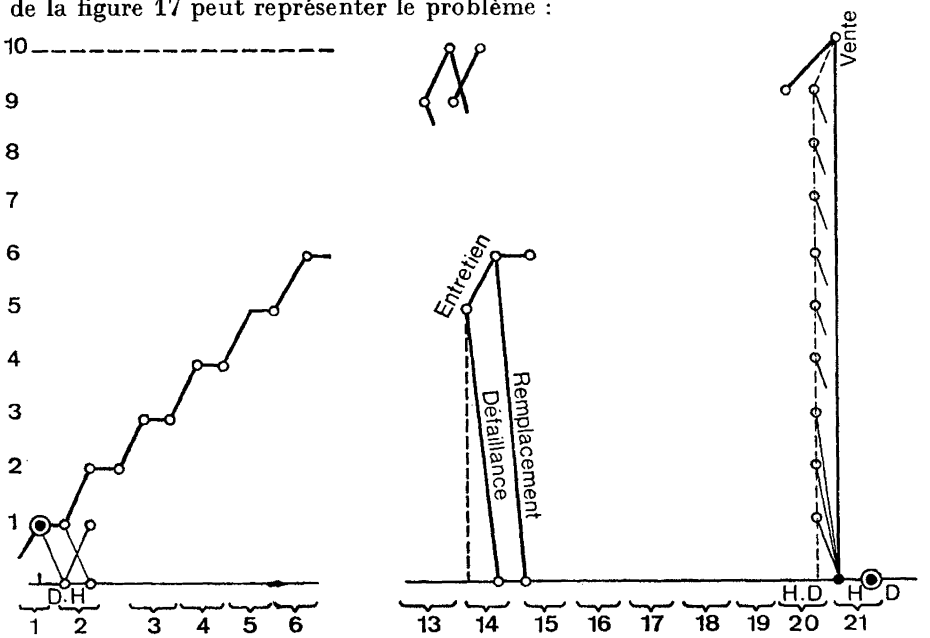

Figure 17

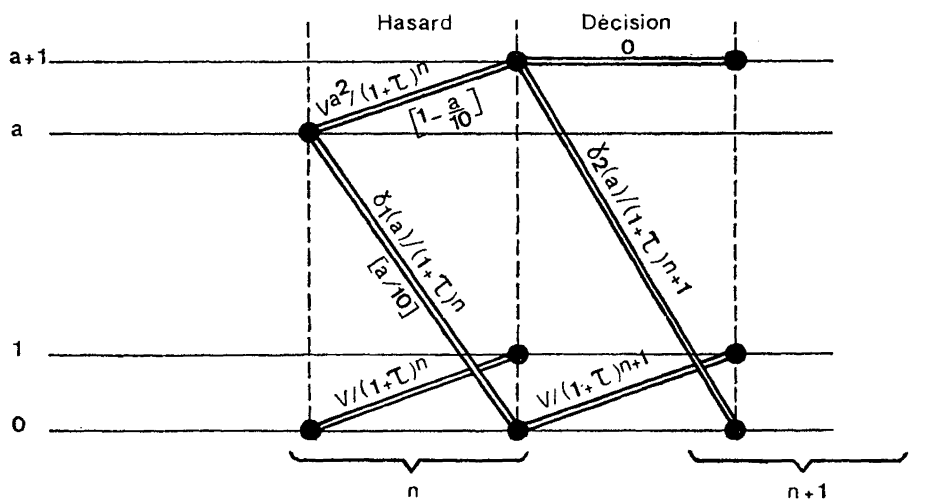

Figure 18

Nous détaillons une partie du graphe en indiquant les coûts et les probabilités associées aux arcs (fig. 18 et fig. 19 pour la fin du processus).

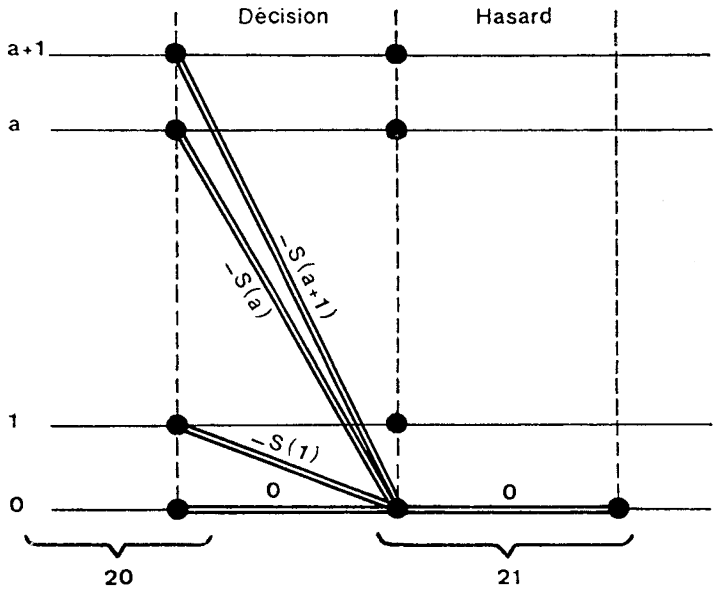

Figure 19

Exemple de problème (D.H.A.)

Soit à gérer pendant 12 mois, un certain stock S. Chaque mois l, le coût d'achat d'une quantité x est $\varphi(l, x)$ et le coût de stockage de S est $\Phi(l, S)$. La demande d est aléatoire : $F(l, d)$ est sa fonction de répartition, le profit réalisé étant $f(l, d)$. Le coût de défaillance est $z(l)$. (On discrétise x, S et d.)

Graphe (fig. 20)

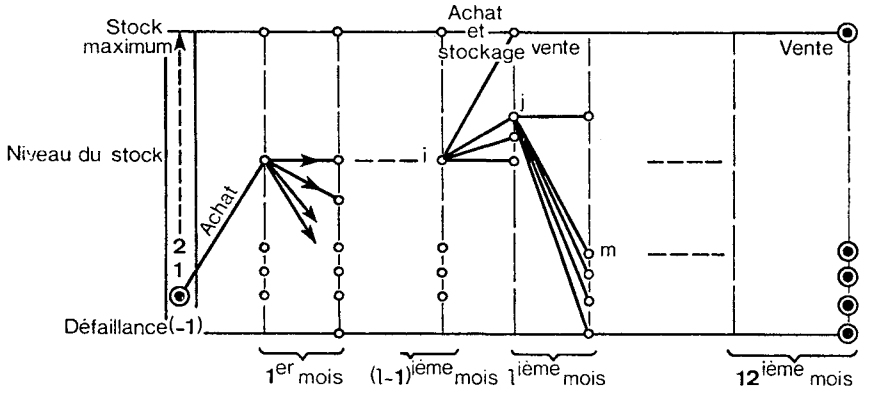

Figure 20

$$\gamma_2(L-1, i, j) = \varphi(L, j-i) + \Phi(L, j) \qquad (j \geqslant i)$$

$$\begin{cases} \gamma_1(L, j, m) = f(L, j-m) & (m \leqslant j) \\ p\ (L, j, m) = F(L, j-m) - F(L, j-m-1) \\ \qquad \text{pour } m \neq -1 \end{cases}$$

$$\begin{cases} \gamma^1(L, j, -1) = f(L, j) - z(L) \\ p\ (L, j, -1) = 1 - F(L, j) \end{cases}$$

5. EXPERIMENTATION SUR ORDINATEUR

Les algorithmes ont été testés sur un ordinateur I.B.M. 360-50. Les graphes traités peuvent être caractérisés par :

$N + 1$: nombre de périodes,
s : nombre de sommets,
a : nombre moyen d'arcs issus d'un sommet.

Prenons le cas extrême où chaque sommet est relié à tous les sommets relatifs à l'instant suivant. On a alors :

Dans le cas déterministe : $Na = s$, nombre total d'arcs : Na^2.
Dans le cas aléatoire : $2Na = s$, nombre total d'arcs : $2Na^2$.

Pour ce qui est indépendant du graphe traité nous avons trouvé :

	Programme seul	Nombre de mémoires occupées	Temps de compilation	Temps de chargement	Temps total de pré-calcul
Déterministe	Alg (K.C.)	$Na(2a + K)$	41″	52″	1′33″
	Alg (R)	$Na(2a + K + 1)$	44″	53″	1′37″
Aléatoire	Alg (K.C.)	$Ka(a + N) + 4a^2N$	63″	44″	1′47″
	Alg (R)	$Na(8a + 3K + 2)$	71″	57″	1′58″

– Les temps de pré-calcul sont comparables pour les deux algorithmes.

– Par contre le nombre de mémoires occupées par Alg (K.C.) est inférieur à celui occupé par Alg (R) dans le cas déterministe. Mais surtout, la figure 7 montre que l'on peut rentrer les données de Alg (K.C.) période par période, ce qui divise le nombre de mémoires occupées par

$\frac{N}{2}$ environ, mais qui allonge considérablement le temps de traitement car la lecture retarde le processus.

EXEMPLE : 50 000 mémoires correspondent à :

Alg (K.C.) — D. —
Alg (R) — D. — } $N = 25$, $a = 30$, $K = 10$

Alg (K.C.) — D.H. — $N = 25$, $a = 20$, $K = 20$
Alg (R) — D.H. — $N = 20$, $a = 15$, $K = 20$.

Donc :

Lorsque le nombre N de périodes devient élevé (pour donner une idée lorsque a^2N atteint l'ordre des dizaines de mille) on doit utiliser Alg (K.C.) en introduisant les données période par période. Le temps de traitement du graphe est alors K fois le temps de recherche de la politique optimale (voir plus loin).

Problème de distorsion de débit : $N = 8$, $s = 150$, $a \sim 20$

Déterministe	Politique 1-optimale	Calcul jusqu'à $K = 10$	Temps de calcul d'un i-optimum $1 \leqslant i \leqslant K$
Alg (K.C.)		65″26	6″30
Alg (R)	8″04	10″62	Entre 0″40 et 0″15 (décroît avec i)

Problème de remplacement de matériel : $N = 20$, $s = 330$, $a \sim 8$

ALÉATOIRE	STRATÉGIE 1 — optimale	CALCUL JUSQU'A $K = 5$	TEMPS DE CALCUL D'UN i-OPTIMUM $1 < i \leqslant K$
Alg (K.C.)		368″58	75″
Alg (R)	59″12	75″86	$i = 2$; 15″68 $i = 3, 4, 5$; 0″38

REMARQUE :

a ne dépasse généralement la dizaine que dans le cas de problème discret par artifice.

Pour les temps de calcul, nous donnons les chiffres obtenus en traitant les problèmes D.A. et D.H.E. du paragraphe 2.

Conclusion

Le graphique de la figure 21 résume les résultats sur la rapidité des deux algorithmes :

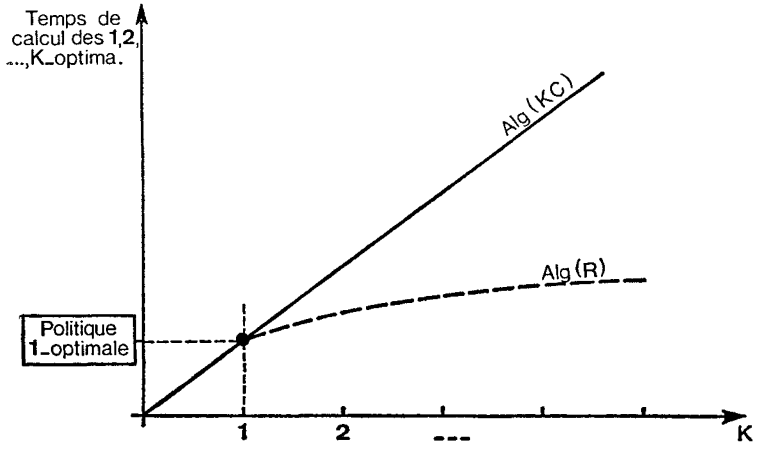

Figure 21

Pour chaque *i*-optimum désiré le temps de calcul est, pour Alg (K.C.) celui du 1-optimum, alors que, pour Alg (R) ce temps est négligeable et décroît. En effet Alg (R) accumule les informations, et le calcul d'un $i + 1$-optimum ne demande que très peu d'opérations, lorsque le *i*-optimum est connu.

Donc :

Lorsque les dimensions du problème le permettent l'exploration du voisinage de l'optimum par l'algorithme à réflexion demande un temps négligeable par rapport au temps decalcul du 1-optimum (le facteur est au moins de quelques dizaines pour chaque *i*-optimum).

BIBLIOGRAPHIE

On trouve la démonstration du théorème de k-optimalité ainsi que l'algorithme alg. (K.C.) dans :

[1] A. KAUFMANN et R. CRUON, « Étude de la sensibilité en programmation dynamique. Politiques k-optimales en avenir certain », *Revue Française de Recherche Opérationnelle*, n° 32 (1964).

[2] A. KAUFMANN et R. CRUON, « Stratégies k-optimales dans les programmes dynamiques stochastiques finis », The Fourth International Conference on Operational Research, Session Advances in Technics of modeling, Boston, 1966.

Sur des problèmes plus généraux, on peut consulter :

[3] R. BELLMAN et R. KALABA, « On k^{th} best policies », *P. Soc. Industr. appl. Math.*, **8**, 4, 582-588, déc. 1960.

[4] R. BELLMAN, *Dynamic Programming*, Princeton University Press (1957).

[5] A. KAUFMANN et R. CRUON, *La programmation dynamique et ses applications*, Dunod (1964).

Enfin, on peut trouver une application effective des notions précédentes à un problème de type (D.A.) dans :

[6] L. THIRIET et A. DELEDICQ, « Applications of suboptimality in Dynamic Programming ». Industrial Engineering Chemistry, vol. 60. N° 2. February 68.