Optimisation dans le domaine de la Sûreté de Fonctionnement

André Cabarbaye
Centre National d'Etudes spatiales (CNES)
18 avenue Edouard Belin
31401 - Toulouse Cedex 4
andre.cabarbaye@cnes.fr
Tél. (33) 5 61 28 27 41
Fax. (33) 5 61 28 22 31

Julien Séroi
Ecole Supérieure d'Ingénieurs en Electronique et Electrotechnique (ESIEE) 2 Bd. Blaise-Pascal
93162 - Noisy-le-Grand Cedex
julien.seroi@wanadoo.fr
Tél. (33) 1 45 92 65 00
Fax. (33) 1 45 92 66 99

Linda Tomasini
Centre National d'Etudes Spatiales (CNES)
18 avenue Edouard Belin
31401 - Toulouse Cedex 4
linda.tomasini@cnes.fr
Tél. (33) 5 61 27 31 73
Fax. (33) 5 61 28 22 31

Résumé
Afin de traiter diverses problématiques d'optimisation dans le domaine de la Sûreté de Fonctionnement, cette communication présente deux techniques apparemment prometteuses : les algorithmes génétiques et la programmation dynamique avec apprentissage par renforcement. Elle s'appuie sur divers travaux menés actuellement dans le domaine spatial.

Introduction
Diverses évaluations de fiabilité/disponibilité sont menées pour dimensionner correctement l'architecture et la mise en œuvre des systèmes [1]. Ces évaluations sont réalisées au moyen de différentes méthodes et outils de modélisation (bloc diagramme de fiabilité, arbre d'événements, graphe de Markov, réseaux de Petri) et de traitement (analytique, markovien, simulation de Monte-carlo) suivant la nature et la complexité des systèmes étudiés.
Mais les besoins évoluent :
. L'évaluation est menée au niveau du système complet, et non plus de ses seuls constituants, afin de maîtriser la disponibilité du service rendu à l'utilisateur final (abonné d'un service de communication par satellites, usager d'un moyen de transport, téléspectateur d'un événement sportif…).
. L'évaluation n'est plus considérée comme une étape de vérification contractuelle, mais est utilisée à des fins d'optimisation pour diminuer le coût global d'acquisition des produits (comprenant le développement, la mise en place, les opérations, la maintenance, le retrait de service...).
Ainsi, l'optimisation de la maintenance, qui a souvent été négligée dans le passé, intéresse de plus en plus les grands donneurs d'ordres (armement, espace, nucléaire….), depuis qu'ils ont pris conscience que le coût du maintien opérationnel d'un produit est généralement supérieur au coût de ce dernier. Suivant la nature des systèmes, l'objectif recherché peut être de minimiser le coût global d'acquisition plus celui induit par les indisponibilités de service, le coût à disponibilité donnée, l'indisponibilité à coût donné ou un ratio entre ces différents paramètres.
Cette optimisation rencontre cependant deux écueils :
. la complexité croissante des systèmes étudiés qui dépendent de nombreux paramètres et présentent généralement plusieurs optima locaux,
. la durée d'évaluation de ces systèmes, notamment quand celle-ci est réalisée par simulation.
Les analyses de sensibilité menées individuellement sur chacun des paramètres montrent vite leurs limites, et il devient nécessaire de faire appel à des techniques d'optimisation plus sophistiquées issues des domaines de la recherche opérationnelle ou de l'intelligence artificielle.
Parmi les pistes explorées, les algorithmes génétiques [2] présentent certains avantages, mais aussi certaines limites qui nous conduisent à proposer une autre méthode, la programmation dynamique avec apprentissage par renforcement [5], pour résoudre la problématique de la maintenance optimale.
Après un bref exposé de diverses problématiques d'optimisation dans le domaine de la Sûreté de Fonctionnement, cette communication présente ces deux techniques en s'appuyant sur des applications du domaine spatial.

1. Des besoins aux méthodes

Les besoins en optimisation sont multiples dans le domaine de la Sûreté de Fonctionnement et recouvrent des enjeux souvent considérables. L'objectif recherché consiste généralement à maximiser, au moindre coût, la robustesse des produits aux aléas (défaillances, erreurs humaines, événements exceptionnels...).
Trois grandes familles de problèmes se présentent aux fiabilistes : l'optimisation de l'architecture des systèmes, l'optimisation de leur mise en œuvre et l'ajustement des paramètres de divers modèles utilisés pour effectuer des évaluations.

1.1. Architecture de systèmes

Un système complexe (satellite, centre de contrôle…) est constitué de nombreux éléments qui peuvent généralement se représenter sous forme d'un Bloc Diagramme de Fiabilité (figure 1).

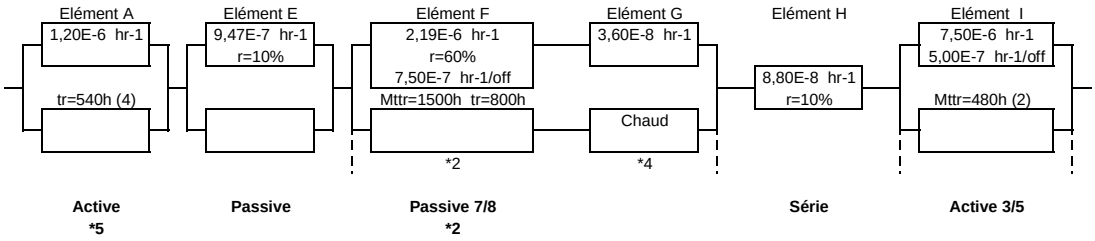


Figure 1 – *Exemple d'architecture de système*

L'architecture d'un tel système résulte de multiples choix portant sur :
. la fiabilité des constituants (taux de défaillance) qui dépend des technologies mises en œuvre,
. leur redondance éventuelle (m parmi n) et les caractéristiques de celle-ci (active ou passive, chaude ou froide, durée de reconfiguration, efficacité de la surveillance et du processus de reconfiguration…),
. leur taux d'utilisation,
. la durée de leur réparation quand ils peuvent être réparés,
. les conditions environnementales....
Ces choix ont tous un impact sur la Sûreté de Fonctionnement du produit (fiabilité, disponibilité, sécurité) et sur son coût. A partir d'un modèle stochastique associant ces deux dimensions, l'optimisation consiste à rechercher une configuration optimale de paramètres divers (réels, entiers ou binaires).

1.2. Mise en œuvre de systèmes

La mise en œuvre d'un système peut recouvrir divers aspects tels que la mise en service, les opérations, la maintenance, le retrait de service… Mais parmi les coûts de mise en oeuvre, celui de la maintenance est souvent prépondérant (figure 2).

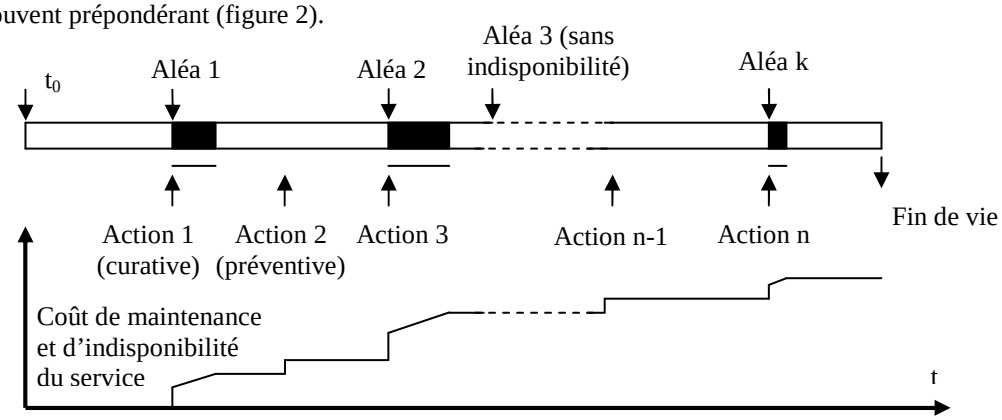


Figure 2 – *Maintenance d'un produit*

L'optimisation de la maintenance consiste à minimiser son coût et celui généré par les indisponibilités du service rendu à l'utilisateur. Elle se ramène à un problème de décisions séquentielles prises pour répondre à des aléas (maintenance curative) ou pour limiter leur occurrence (maintenance préventive).

1.3. Ajustement des paramètres d'un modèle

Afin de pouvoir évaluer les caractéristiques de Sûreté de Fonctionnement d'un système, divers modèles probabilistes sont utilisés pour caractériser les transitions entre les différents états possibles de ses constituants (défaillance, réparation…).

Parmi ceux-ci, la loi de Weibull (à 3 paramètres) est souvent employée pour effectuer des simulations de Monte-Carlo. Mais des modèles plus complexes sont parfois nécessaires notamment dans le cadre de traitements markoviens. Ainsi, la méthode des états fictifs consiste à remplacer dans un modèle markovien une transition quelconque entre deux états par un ensemble de transitions à taux constants entre des états fictifs (figure 3).

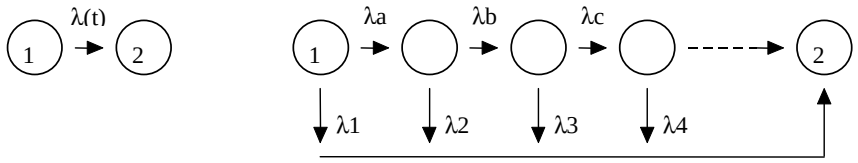


Figure 3 – *Méthode des états fictifs (modèle de Cox)*

Cette méthode permet de modéliser des durées de transition pseudo-déterministes ou des taux complexes (courbe en baignoire considérant les pannes de jeunesse et les phénomènes d'usure), en se ramenant à un système markovien homogène (à taux constants) que l'on sait traiter. Elle peut être également utilisée pour pallier l'explosion combinatoire des états d'un système en remplaçant des modèles de sous-ensembles par des modèles simplifiés équivalents limités à quelques états. Mais la principale difficulté de cette méthode réside dans l'ajustement des taux de transition du modèle qui est choisi en fonction de la complexité de la loi de transition (modèle de Cox de degré k ou d'Erlang généralisé avec $\lambda 1 = \lambda n$).

Quel que soit le modèle utilisé, l'optimisation consiste à rechercher une configuration optimale de paramètres (réels) en utilisant la méthode des moindres carrés, si l'ajustement est réalisé à partir d'une courbe de fiabilité, ou la méthode du maximum de vraisemblance si celui-ci est réalisé à partir d'un échantillon de durées de transition (statistique de durées de réparation par exemple).

1.4. Choix parmi les techniques d'optimisation

Nous n'avons pas la prétention de présenter ici l'ensemble des techniques d'optimisation mais simplement d'indiquer certaines caractéristiques des méthodes les plus connues afin de chercher à répondre à notre besoin (figure 4).

Méthodes locales (déplacement dans l'espace des états) :

. Arrêt au premier optimum local : *gradient, simplexe ...*

. Exploration dans tout l'espace des états : *algorithmes génétiques, recuit simulé, méthode tabou...*

Méthodes globales (sensées trouver l'optimum global) :

analyse par intervalles, optimisation combinatoire, programmation dynamique…

Figure 4 – *Méthodes d'optimisation*

Parmi ces méthodes, les algorithmes génétiques apparaissent bien adaptés à l'optimisation des architectures de systèmes et à l'ajustement des paramètres de modèles. Ils ne nécessitent pas la connaissance analytique de la fonction à optimiser et peuvent être couplés à des outils d'évaluation considérés comme des boîtes noires (les fonctions d'évaluation présentent généralement plusieurs optima qui ne peuvent pas être tous atteints par une simple méthode de grimpeur tel que le gradient ou le simplexe). Les algorithmes génétiques apparaissent cependant mal adaptés aux problèmes de maintenance bien que l'on puisse envisager d'optimiser les paramètres d'une stratégie de maintenance définie a priori qui ne soit pas forcément optimale.

Les techniques de programmation dynamique semblent, par contre, bien convenir à ce type de traitement, car elles ont justement pour objet de résoudre les problèmes de décisions séquentielles.

2. Les algorithmes génétiques

Développés par John Holland et ses collaborateurs à l'université du Michigan, les algorithmes génétiques sont des algorithmes d'optimisation fondés sur les mécanismes de la sélection naturelle et de la génétique. Le premier de ces mécanismes tient aux principes de la survie des espèces les mieux adaptées fondé sur le postulat de Darwin. Le second s'appuie sur la diversité des individus dans la population d'une même espèce, qui évolue au cours du temps par des croisements ou mutation.

2.1. Présentation des algorithmes génétiques

L'analogie entre la biologie et les algorithmes génétiques est présentée dans la figure 5.

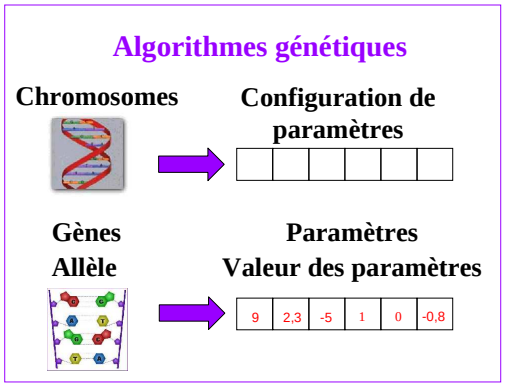


Figure 5 - *Analogie entre la biologie et les algorithmes génétiques*

Chaque configuration de paramètres correspond à un chromosome dont les gènes sont des paramètres de différents types (binaire, entier, réel, éventuellement borné). Ces chromosomes subissent au sein d'une population des opérations de mutation, croisement, sélection… tenant compte des performances respectives de chacun (figure 6).

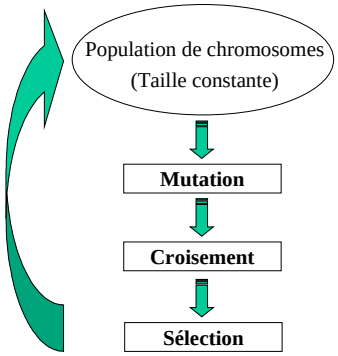


Figure 6 - *Principe de base des algorithmes génétiques*

A chaque génération, une nouvelle population de même taille est créée, constituée en partie des meilleurs éléments de la génération précédente et d'éléments nouveaux générés par mutation ou croisement. Ces opérations sont menées suivant deux objectifs : atteindre les optima locaux et explorer l'espace des variables pour rechercher l'ensemble des optima afin de trouver par là même l'optimum global.

La **mutation** consiste à introduire un bruit dans la valeur d'un gène d'un chromosome, c'est à dire un écart aléatoire autour de cette valeur. En ce sens la mutation est une opération **d'exploration** de l'espace de recherche. La figure 7 présente un exemple de mutation applicable à différents types de paramètres.

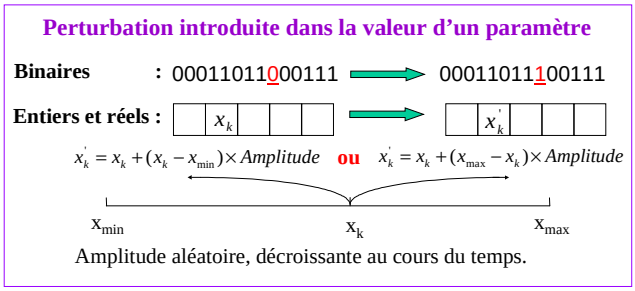


Figure 7 - *Exemple de mutation*

Dans cet exemple, la mutation d'un chromosome tiré aléatoirement dans la population s'effectue par modification de l'un de ses gènes choisi au hasard. Celui-ci change simplement d'état s'il est binaire ou réalise un saut d'amplitude décroissante au cours du temps s'il est réel ou entier, afin de limiter progressivement l'exploration au fur et à mesure de la recherche.

Le **croisement** est effectué en appariant deux chromosomes de la population qui s'échangent de l'information entre eux pour donner naissance à deux fils. De même que la mutation, le croisement est une opération **d'exploration** de l'espace de recherche dont deux exemples sont proposés à la figure 8.

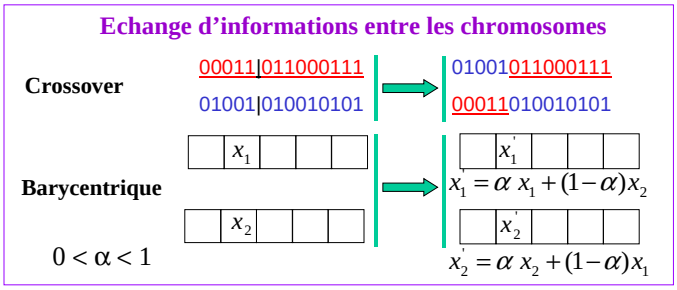


Figure 8 - *Exemples de croisement*

Dans ces exemples, le croisement de deux chromosomes parents tirés aléatoirement dans la population s'effectue soit par échange de gènes (crossover), chacun des gènes étant reproduit chez l'un ou l'autre des fils, soit en moyennant les valeurs (entiers ou réels) des gènes des parents (barycentrique).

La **sélection** est un procédé par lequel chaque chromosome est copié un certain nombre de fois dans la nouvelle population en fonction de la valeur (ou fitness) de la fonction à optimiser (appelée aussi fonction d'adaptation). Les chromosomes dont la valeur de la fonction d'adaptation est élevée ont une forte probabilité de contribuer à la génération suivante, en créant un ou plusieurs descendants identiques à eux-mêmes. Cet opérateur, dont un exemple est proposé à la figure 9, est bien entendu une version artificielle de la sélection biologique. Dans la nature, l'adaptation d'une espèce est déterminée par sa capacité à survivre aux prédateurs, aux maladies, et aux obstacles à franchir pour atteindre l'âge adulte et la période de reproduction, alors que dans notre environnement artificiel, la fonction à optimiser est l'arbitre final de la vie ou de la mort de chaque chromosome. En ce sens l'opération de sélection est une opération **d'exploitation** de l'espace de recherche.

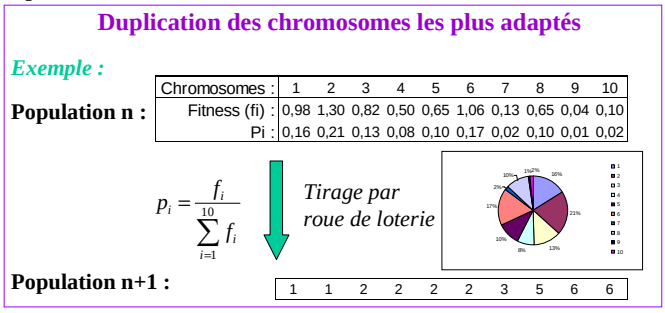


Figure 9 - *Exemple de sélection*

Dans cet exemple, la probabilité de sélection pi de chaque chromosome, calculée à partir du poids relatif du résultat de son évaluation, correspond à un secteur de roue de loterie avec lequel on effectue N tirages pour obtenir la nouvelle population (N étant la taille constante de la population).

Outre les exemples indiqués, les opérations de mutation, croisement et sélection peuvent être réalisées de multiples manières qui se révèlent plus ou moins efficaces suivant les problèmes à traiter. Par ailleurs, la recherche de l'optimum peut être améliorée en couplant à ces opérations de base des techniques de mise à l'échelle, d'élitisme, ou d'optimisation plus classiques. La mise à l'échelle est une transformation agissant sur la valeur de la fonction d'adaptation qui a pour but de créer un effet de zoom sur les résultats au fur et à mesure de la recherche. Au début de la recherche, on veut réduire les écarts entre les fitness afin d'éviter que les bons chromosomes deviennent trop prépondérants. Puis on amplifie les écarts pour accélérer la convergence (figure 10). L'élitisme consiste à conserver à chaque génération un certain nombre des meilleurs chromosomes de la population qui pourraient disparaître par les opérations de mutation, croisement ou sélection. Une méthode de grimpeur, telles que les méthodes du gradient ou du simplexe, peut être également associée à l'algorithme génétique pour constituer ensemble une méthode hybride ayant une meilleure capacité d'exploitation. Toutefois, la méthode du simplexe est généralement préférée à celle du gradient car elle n'impose pas de calcul de dérivée.

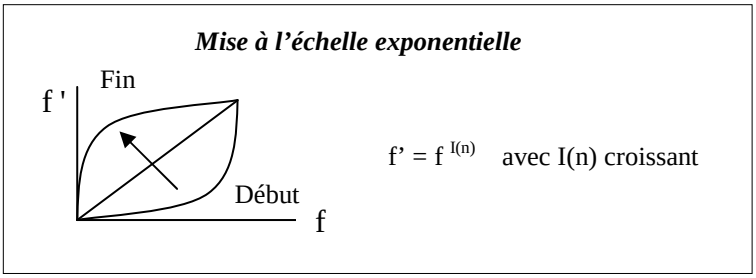


Figure 10– *Exemple de mise à l'échelle*

2.2. Mise en œuvre des algorithmes génétiques [7]

Diverses problématiques nous ont conduits à développer un outil générique d'optimisation par algorithmes génétiques pouvant s'adapter à différents outils d'évaluation (traitement markovien, simulation de Monte-Carlo...). Développer sous Excel™, celui-ci intègre plusieurs opérateurs de mutation, croisement et sélection ainsi que diverses améliorations (mise à l'échelle, élitisme et simplexe), que l'on peut choisir et paramétrer suivant les problèmes à traiter. Son principe est décrit à la figure 11.

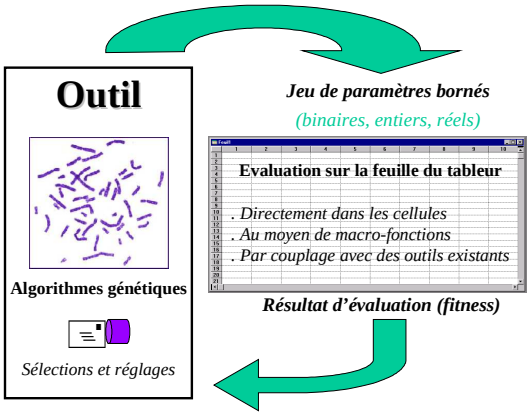


Figure 11 – *Outil générique d'optimisation par algorithmes génétiques*

Cet outil a permis d'optimiser certaines architectures (satellites et stations de contrôles) et d'ajuster des paramètres de modèles markoviens. Il a été également utilisé pour minimiser la durée de déploiement d'une constellation de 32 satellites (voir figure 15), mis à poste sur 8 plans orbitaux via une orbite de dérive, en jouant sur les types de lanceurs (de 2, 4 ou 6 satellites), les dates de tir, et la répartition des

satellites sur les lanceurs. Il a ainsi permis de trouver une séquence de tir meilleure que toutes celles qui avaient pu être imaginées auparavant (12 mois au lieu de 14).
Mais cette mise en œuvre a aussi permis de soulever certaines difficultés et limitations.
L'un des premiers obstacles rencontrés réside dans l'interdépendance des paramètres qui peut être difficilement prise en compte au niveau d'algorithmes génétiques qui se veulent génériques. Cette difficulté peut cependant être levée par élimination du domaine de recherche où s'exercent les contraintes au moyen de l'opérateur de sélection (en affectant à ce domaine un mauvais résultat d'évaluation), ou par un judicieux changement de variables comme celui décrit dans l'exemple du voyageur de commerce de la figure 12. Les contraintes liées à la répartition des satellites dans la constellation ont ainsi été éliminées.

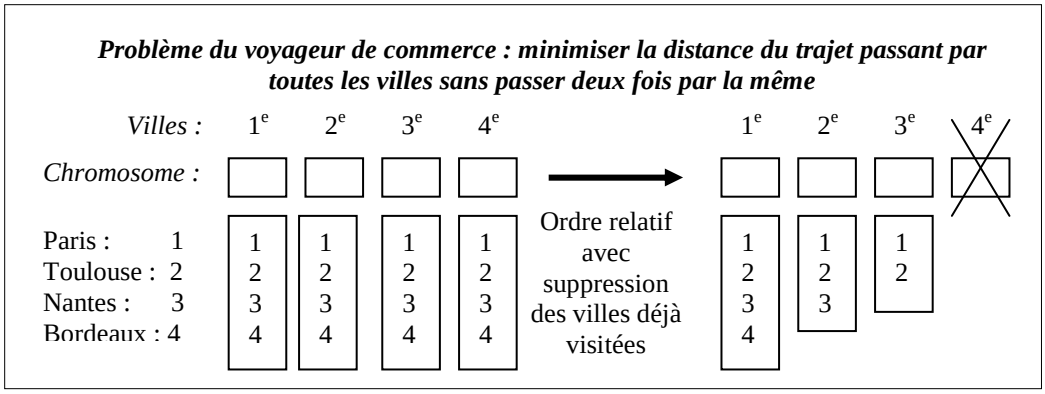


Figure 12 – *Suppression de contraintes par changement de variables*

La durée de l'évaluation est souvent rédhibitoire, notamment quand celle-ci est réalisée par simulation de Monte-Carlo, même si certains palliatifs peuvent être envisagés. L'optimisation peut s'effectuer en deux temps. Un optimum grossier est d'abord obtenu au moyen d'évaluations peu précises, puis un optimum plus fin est recherché autour du premier au moyen d'évaluations plus fines. Dans certains cas, il est également possible d'effectuer beaucoup plus rapidement l'évaluation grossière au moyen d'un réseau de neurones [3] dont l'apprentissage a été réalisé préalablement à partir d'un certain nombre de résultats d'évaluation.
Enfin la convergence des algorithmes génétiques vers un optimum global n'est jamais assurée et cela notamment quand certains des paramètres utilisés sont des entiers (les opérateurs de mutation et de croisement effectuent alors des troncatures). Cette convergence n'est pas démontrée à ce jour sur le plan théorique dans le cas général.

3. Programmation dynamique avec apprentissage par renforcement

La programmation dynamique permet d'optimiser une séquence de décisions. Cette technique est basée sur une modélisation mathématique sous forme de Processus Décisionnel de Markov (PDM).

3.1. Processus Décisionnel de Markov

Un processus décisionnel de Markov est un modèle du comportement d'un agent qui au cours du temps (discrétisé) adapte ses décisions en fonction de l'état courant du système qu'il a en charge de contrôler [6].
Un tel processus représenté en figure 13 s'appuie sur trois concepts :

- le graphe de Markov qui représente les états du système (sommets e_i) et les possibles transitions entre eux (arcs orientés).
- les actions ($a_{i,k}$) que peut réaliser l'agent dans chacun des états (e_i). La conséquence d'une action n'est pas nécessairement déterministe et la transition d'un état e_i vers un état e_j après une action $a_{i,k}$ est régie par la probabilité $p(e_j / e_i, a_{i,k})$ entre t et t+1. La notion d'action conduit à définir celle de politique : on appelle politique la fonction π qui à tout état e_i associe une action a_i.
- les récompenses $r(e_j / e_i, a_{i,k})$ ou coûts qui pour chaque transition résultent du triplet (e_i, $a_{i,k}$, e_j). La politique optimale est la politique qui maximise en moyenne les revenus (ou minimise les coûts).

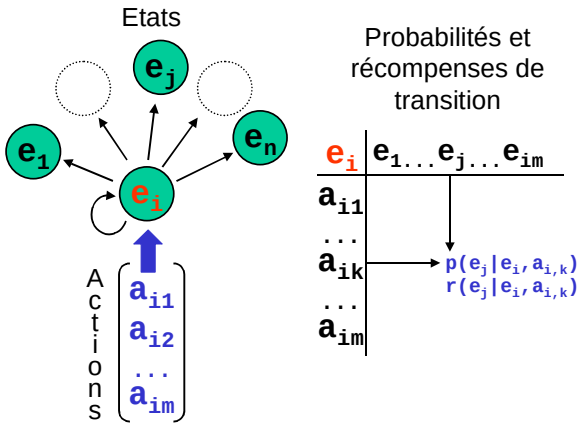


Figure 13 – *Exemple de Processus décisionnel de Markov*

3.2. La programmation dynamique

Proposée par Bellman, la programmation dynamique repose sur le principe suivant : dans une séquence optimale de décisions, la séquence restante, après chaque décision, forme une sous séquence optimale. Ce qui revient à dire que la programmation dynamique cherche à résoudre l'ensemble de tous les sous problèmes du problème posé.

Appliqué à un PDM, il s'agit de connaître la meilleure action à réaliser quel que soit l'état où l'on se trouve.

Mais lorsque l'on cherche à résoudre un problème où l'espace d'états est très grand, la programmation dynamique, relativement lourde en terme de complexité algorithmique, ne permet pas de trouver de solution en un temps raisonnable ou nécessite une capacité mémoire prohibitive. De plus, lorsque le modèle dynamique est mal connu, la résolution par cette méthode n'est plus possible.

Cependant, les techniques d'apprentissage par renforcement [5], ou *neuro-dynamic programming*, permettent de repousser ces limites, en exploitant un modèle de simulation du système étudié.

3.3. L'apprentissage par renforcement

L'apprentissage par renforcement consiste à remplacer l'ensemble du modèle décisionnel de Markov par un simulateur donnant à partir d'un état courant ei et une action aik l'état suivant ej et le revenu associé $r(e_j / e_i, a_{i,k})$, comme le montre la figure 14.

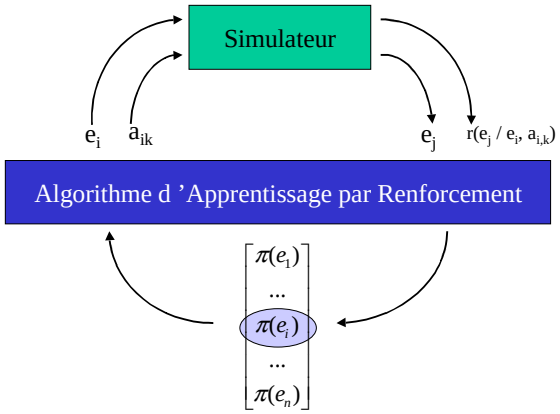


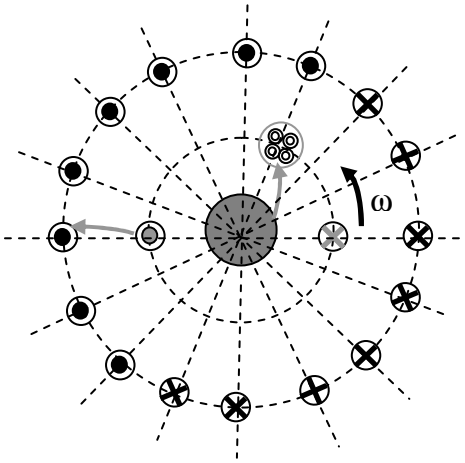
Figure 14 – Principe général de l'apprentissage par renforcement

A partir d'un état initial e_0 et d'une politique π_0 arbitraire, on effectue une simulation du comportement du système tout en cherchant à améliorer au fur et à mesure la politique suivant les revenus obtenus.

Ainsi, la politique optimale est déterminée de manière progressive en mettant à jour la fonction π, en priorité pour les états les plus probables et donc les plus rencontrés dans la simulation.
Pour ne pas s'enferrer dans une politique non optimale, en raison de combinaisons de transitions à forte probabilité, deux techniques d'exploration peuvent être utilisées :
- certaines décisions sont prises de façon aléatoire durant la simulation, sans tenir compte de la politique en cours (cette technique est même nécessaire pour assurer la convergence de l'algorithme),
- différentes simulations sont réalisées en faisant varier les états initiaux.
Si ces techniques d'explorations ne permettent pas d'assurer que la politique obtenue soit optimale pour tous les états rares, dans un temps raisonnable, il est toujours possible d'optimiser a posteriori la décision pour un état donné en le prenant systématiquement comme état initial.

3.4. Mise en œuvre

La programmation dynamique avec apprentissage par renforcement [4] fait actuellement l'objet d'une application relativement complexe, afin d'évaluer son apport à la problématique de la maintenance optimale [5, 6]. Celle-ci consiste à optimiser le coût de l'entretien et du renouvellement d'une constellation de 32 satellites (figure 15) à partir d'un modèle stochastique considérant la fiabilité et le coût des satellites et des lanceurs, la durée de vie des satellites (liée à l'usure et aux ergols) et le coût de la dégradation du service induite par le nombre et la position de satellites indisponibles.



. 4 satellites opérationnels par plan sur 8 plans

. Injection par lanceur de grappe de satellites sur orbites de dérive puis injection autonome sur orbites finales

. Rotation d'un plan par mois des orbites de dérive

. Capacité des lanceurs de 2, 4 ou 6 satellites

Figure 15 – *Constellation de satellites vue de dessus (injection sur orbite de dérive)*

La résolution d'un tel problème nécessite la validation d'un certain nombre d'étapes successives :

a) formalisation du problème

En premier lieu, il convient de modéliser le système étudié en adoptant une représentation synthétique des états et des actions, puis en définissant sa dynamique propre [5, 6]. Il est, en effet, impossible d'énumérer tous les états de la constellation et les actions possibles. Chaque état du système a donc été défini comme une combinaison des états individuels de toutes les entités qui constituent la constellation et son dispositif de maintenance.
L'état de la constellation se présentent ainsi sous forme d'une liste de la manière suivante :
$e = (e^{op}, e^{sp}, e^{sol})$ où

$e^{op} = (x_1, x_2, \ldots x_{32})$ avec x_i la durée de vie restant avant la fin de vie du satellite opérationnelle i ($x_i = 0$ si i en panne)

e^{sp} une liste similaire définissant les satellites de rechange sur des orbites de dérive

e^{sol} une liste définissant les satellites au sol attendant d'être lancés

De même les actions possibles se présentent sous forme d'une liste de la manière suivante :
$a = (a^{sp}, a^{sol}, a^{lanc})$ où

a^{sp} une liste de transferts de satellites d'orbites de dérive vers des orbites opérationnelles

a^{sol} une liste de décisions de tir de satellites

a^{lanc} une liste de choix d'orbites de dérive

Cette étape est aujourd'hui achevée.

b) Réalisation du simulateur

Cette étape est en cours.

c) Implémentation de l'apprentissage par renforcement

d) Réalisation d'un dispositif de stockage de la politique courante

Un réseau de neurones peut être utilisé pour approximer la fonction π et minimiser par là même la capacité de stockage.

Remarque :
Les algorithmes génétiques auraient pu être envisagés pour traiter ce type de problème en s'appuyant sur une stratégie de renouvellement paramétrable, définie a priori. Mais cette stratégie n'aurait pas été forcément optimale. Ainsi le renouvellement d'un satellite peut s'effectuer en cas de défaillance de celui-ci ou peu avant la date prévue de sa fin de vie. Mais il peut être intéressant de remplacer précocement un satellite ayant encore un certain potentiel après la défaillance d'un satellite opérant sur une orbite voisine, afin de bénéficier d'un lancement multiple diminuant sensiblement le coût d'un satellite à poste.

4. Conclusions

L'optimisation dans le domaine de la Sûreté de Fonctionnement recouvre des enjeux considérables qui n'ont pas toujours été évalués par le passé à leur juste mesure. Une nette prise de conscience s'observe cependant aujourd'hui chez la plupart des grands donneurs d'ordres (armement, espace, nucléaire….).
Cette optimisation rencontre cependant trois difficultés : l'obtention de données de fiabilité pertinentes, la disponibilité de modèles paramétriques de coût et le faible héritage sur le plan méthodologique, outre les techniques classiques d'évaluation.
Cette contribution a tenté d'apporter une réponse à cette dernière difficulté en proposant deux techniques d'optimisation apparemment bien adaptées à la Sûreté de Fonctionnement : les algorithmes génétiques et la programmation dynamique avec apprentissage par renforcement. Elle repose cependant sur des travaux qui ne sont pas totalement achevés. Ces derniers recouvrent également d'autres techniques, telles que le recuit simulé ou la méthode tabou, et feront l'objet de publications ultérieures.

Références :
[1] A. Cabarbaye, *Modélisation et évaluation des systèmes* - Cours de technologies spatiales, Edition Cepadues Toulouse 1998
[2] David E. Goldberg, *Algorithmes Génétiques, Exploration optimisation et apprentissage automatique*, Addison-Wesley, 1994.
[3] J-M. Renders, *Algorithmes génétiques et réseaux de neurone*, Hermes, 1995
[4] R. S. Sutton , A. G. Barto, *Reinforcement Learning : An Introduction*, MIT Press, Cambridge, MA, 1998.
[5] F. Garcia, *Rapport d'avancement : Optimisation de systèmes complexes en Sûreté de Fonctionnement* INRA Rapport UBIA N° 1998/4 déC 98.
[6] A. Cabarbaye - F. Garcia - L. Tomasini, *Apport de l'apprentissage par renforcement aux problèmes de maintenance optimale : application aux constellations de satellites, Congrès ROADEF ' 99.*
[7] L. Tomasini, A. Cabarbaye, S. Allibe - *Apport des algorithmes génétiques à la Sûreté de Fonctionnement et à l'optimisation des systèmes, Congrès Qualita 99, 25-26 mars 99 Paris.*