

Optimisation de la maintenance d'une constellation de satellites

Julien Séroi, André Cabarbaye, Linda Tomasini
Centre National d'Etudes spatiales (CNES)
18 avenue Edouard Belin, 31401 Toulouse
Tél. 05 61 28 27 41 - Fax. 05 61 28 22 31
E-mail : julien.seroi@wanadoo.fr
andre.cabarbaye@cnes.fr
linda.tomasini@cnes.fr

Frédéric Garcia
Institut de Recherche Agronomique
(INRA), Unité de Biométrie et Intelligence
Artificielle
B.P. 27, 31326 Castanet-Tolosan
Email : fgarcia@toulouse.inra.fr

Résumé

Cet article présente des travaux de Recherche & Développement menés au Centre Spatial de Toulouse en collaboration avec le laboratoire de Biométrie et d'Intelligence Artificielle de l'Institut de Recherche Agronomique (INRA). Ils concernent l'optimisation de la maintenance d'une constellation de satellites et s'intéressent à une technique d'optimisation peu utilisée à ce jour, la Programmation Dynamique avec Apprentissage par Renforcement, qui semble particulièrement bien adaptée à la problématique de la maintenance.

Introduction

Diverses constellations de satellites se développent aujourd'hui pour assurer des services de télécommunication de couverture mondiale (figure1). De plus en plus sophistiqués, ces projets au coût vertigineux mettent en œuvre de très nombreux satellites (GPS : 22; Iridium : 66 , Skybridge : 64, Globalsar : 48, Teledesic : 288, Celestri : 79).

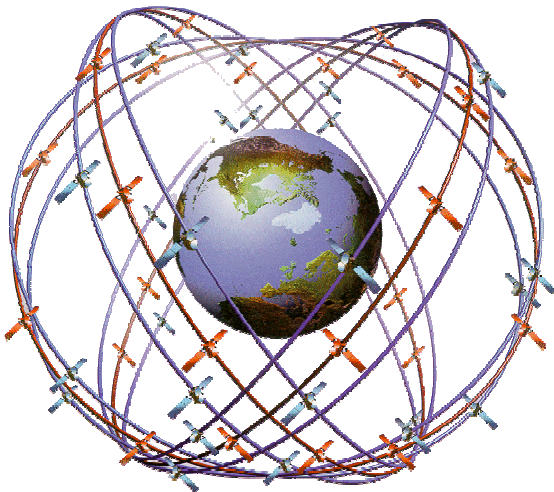


Figure 1 - Constellation de satellites

Outre les choix d'architecture (type et nombre de satellites, orbites utilisées, moyens au sol mis en œuvre...), les politiques de maintenance envisagées ont un impact très significatif sur le coût d'exploitation de tels systèmes. En effet, ce coût recouvre le remplacement des satellites en panne (ce qui est relativement fréquent en raison de leur

nombre) et celui engendré par d'éventuelles indisponibilités temporaires du service rendu aux utilisateurs.

Mais si l'optimisation de la maintenance représente ici un enjeu considérable, celle-ci est généralement très difficile à réaliser en raison de l'extrême complexité des systèmes étudiés.

Les problèmes de maintenance optimale (figure 2) sont des problèmes de décisions séquentielles prises pour répondre à des aléas (actions de maintenance curative) ou pour limiter leur occurrence (maintenance préventive). Dans le cas d'une constellation, la maintenance consiste principalement¹ à lancer des satellites par grappes, au moyen de lanceurs de capacités variables, pour remplacer des satellites défectueux ou proches de leur fin de vie opérationnelle (limitation liée à l'usure et à la capacité d'ergol) ou pour placer en orbite des satellites de rechange afin de limiter les durées de reconfiguration.

Une première approche consiste à effectuer une modélisation du système intégrant une politique de maintenance paramétrable définie a priori, puis de rechercher les paramètres optimaux de cette politique (décision de remplacer systématiquement chaque satellite de la constellation un certain temps avant sa fin de vie prévue, par exemple). Cette recherche peut alors s'opérer soit analytiquement pour des modèles très simples, soit numériquement par une analyse de sensibilité menée individuellement sur chacun des paramètres ou par l'emploi de techniques d'optimisation plus sophistiquées tels que les Algorithmes Génétiques, la recherche Tabou ou le recuit simulé.

Mais les résultats de cette approche dépendent beaucoup de l'expérience de l'analyste car la

¹ La maintenance concerne également d'éventuelles reconfigurations entre équipements en redondance d'un même satellite.

politique initialement choisie n'est pas forcément la meilleure.
 Une seconde approche cherche à prendre véritablement la décision optimale dans chacun des états du système.

Le graphe de Markov représente les différents états du système (sommets e_i) et les possibles transitions entre eux (arcs orientés).
 Dans chacun des états (e_i), l'agent peut réaliser l'une des actions (a_{ik}), qui comprend notamment l'absence d'intervention sur le système.

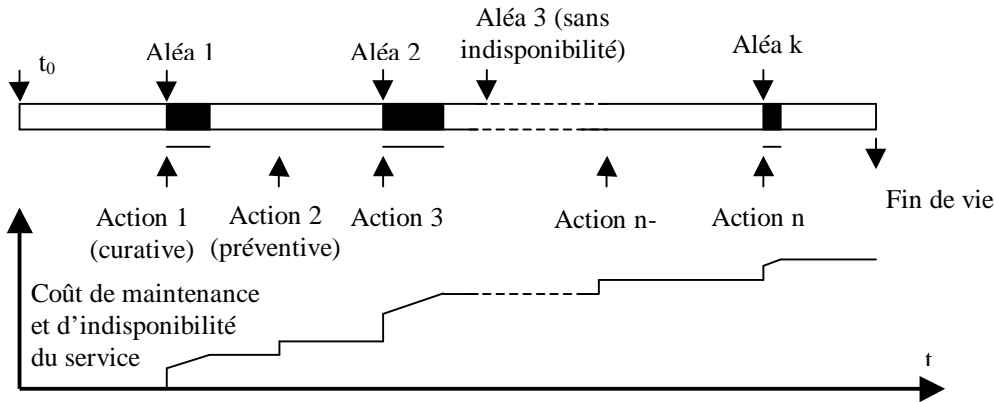


Figure 2 – Maintenance d'un produit

Afin d'optimiser l'exploitation de divers systèmes spatiaux, et plus généralement de pouvoir traiter correctement la problématique de la maintenance optimale, le laboratoire de Sécurité de Fonctionnement du Centre National d'Etudes Spatiales de Toulouse s'est associé avec le laboratoire de Biométrie et d'Intelligence Artificielle de l'Institut de Recherche Agronomique (INRA). En effet, ce laboratoire toulousain est l'un des rares en France à travailler sur une technique d'optimisation originale, la **Programmation Dynamique avec Apprentissage par Renforcement** [2], qui semble particulièrement bien adaptée à ce type de problématique.

En effet, la Programmation Dynamique est une technique assez largement diffusée qui a pour objet de résoudre les problèmes de décisions séquentielles. Son domaine d'application est relativement limité mais peut être considérablement élargi par la technique de l'Apprentissage par Renforcement.

Un cas d'application représentatif, relativement complexe, a été choisi pour évaluer l'apport et les limites de ces méthodes.

Programmation Dynamique avec Apprentissage par Renforcement

La Programmation Dynamique

Proposée par Bellman, la Programmation Dynamique cherche à optimiser une séquence de décisions. Elle se base sur une modélisation mathématique du comportement d'un agent qui adapte ses décisions au cours du temps, en fonction de l'état courant du système qu'il a en charge de contrôler.

Cette modélisation se présente sous la forme d'un Processus Décisionnel de Markov (figure 3).

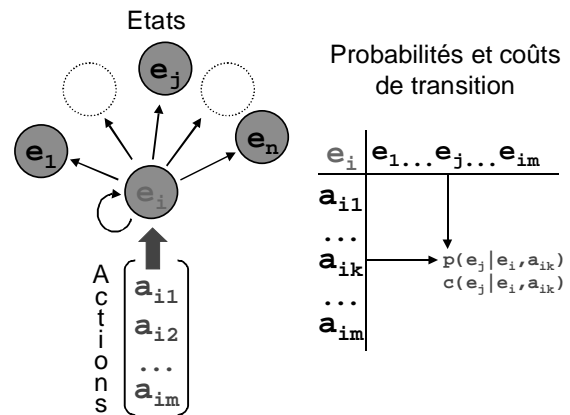


Figure 3 – Processus Décisionnel de Markov

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_{ik} est régie par la probabilité $p(e_j / e_i, a_{ik})$ entre t et $t+1$.

Un coût $c(e_j / e_i, a_{ik})$, qui peut être négatif et donc une récompense, sanctionne chaque action et résulte du triplet (e_i, a_{ik}, e_j) .

La notion d'action conduit à définir celle de politique, c'est-à-dire la fonction π qui à tout état e_i associe une action a_i .

La politique optimale est celle qui minimise en moyenne les coûts (ou maximise les revenus). Elle conduit à réaliser la meilleure action quel que soit l'état où l'on se trouve.

La Programmation Dynamique permet de rechercher cette politique optimale parmi toutes les politiques possibles. Elle repose sur la notion de valeur attribuée à chaque état qui est définie par l'équation de Bellman dans les cas déterministes [1] ou indéterministes [2].

$$V^*(e) = \min_a \{c_{(e|e,a)} + V^*(e')\} \quad [1]$$

$$V^*(e) = \min_a \left\{ \sum_{e'} p_{(e'|e,a)} [c_{(e|e,a)} + V^*(e')] \right\} \quad [2]$$

Cette valeur correspond à la somme des coûts qui seront générés dans le futur à partir de cet état, en considérant que toutes les actions prises par l'agent seront optimales.

Dans le cas où les coûts ne sont engendrés que par les actions prises par l'agent l'équation [2] peut se simplifier [3].

$$V^*(e) = \min_a \left\{ c_{(e|e,a)} + \sum_{e'} p_{(e'|e,a)} V^*(e') \right\} \quad [3]$$

Et si le problème n'est pas à horizon fini ou concerne une mission de relativement longue durée, il convient de faire intervenir un facteur de pondération ($\gamma < 1$) dans l'expression [3] afin de privilégier les coûts (ou revenus) les plus immédiats [4].

$$V^*(e) = \min_a \left\{ c_{(e|e,a)} + \gamma \sum_{e'} p_{(e'|e,a)} V^*(e') \right\} \quad [4]$$

Dans chacun des états, la meilleure action à prendre est alors celle qui conduit le système dans un état de plus faible valeur, au moindre coût. La politique optimale est ainsi définie par l'équation [5] qui peut être résolue par une méthode itérative.

$$a^*(e) = \arg \min_a \left\{ c_{(e|e,a)} + \lambda \sum_{e'} p_{(e'|e,a)} V^*(e') \right\} \quad [5]$$

Mais lorsque l'espace d'états est trop grand, cette résolution ne peut pas être réalisée en un temps raisonnable ou nécessite une capacité mémoire prohibitive.

L'Apprentissage par Renforcement

L'Apprentissage par Renforcement permet de repousser ces limites en remplaçant l'ensemble du modèle décisionnel de Markov par un simulateur donnant à partir d'un état courant e_i et une action a_{ik} l'état suivant e_j et le revenu associé $r(e_j / e_i, a_{ik})$, comme le montre la figure 4.

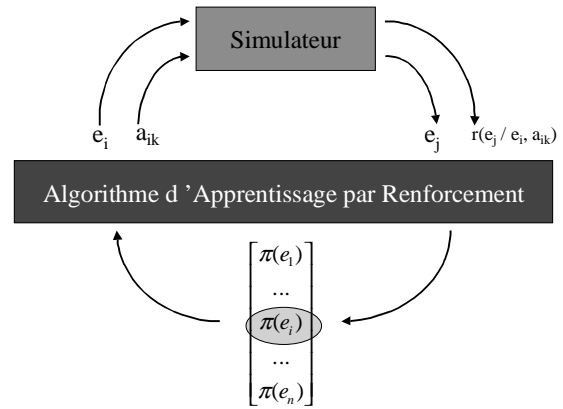


Figure 4 – Principe de l'Apprentissage par Renforcement

A partir d'un état initial e_0 et d'une politique π_0 choisie arbitrairement, on peut alors effectuer une simulation du comportement du système en cherchant à améliorer progressivement la politique employée, suivant les revenus obtenus.

Un algorithme d'apprentissage est utilisé pour améliorer au fur et à mesure cette politique courante. En effet, la valeur des états ne peut plus être calculée mais est apprise en capitalisant l'expérience acquise à chaque pas de simulation.

Mais pour des problèmes aussi complexes que les constellations, le nombre très élevé d'états du système rend le stockage de ces valeurs impossible. Une technique palliative existe cependant. Elle consiste à utiliser une fonction de valeur caractérisant un nombre restreint de paramètres représentatifs de l'état du système. Une telle paramétrisation ne permet plus alors d'espérer trouver la politique optimale, applicable dans chacun des états du système, mais seulement de s'en approcher. Cette fonction de valeur "approchée" peut être réalisée au moyen d'un réseau de neurones ou tout simplement à l'aide d'une combinaison linéaire entre les paramètres choisis.

Pouvant attribuer une valeur à chaque état du système il devient alors possible de suivre une politique qui se rapproche progressivement de la politique optimale, notamment pour les états les plus couramment rencontrés dans la simulation.

Diverses techniques d'exploration permettent d'activer artificiellement des états moins courants, et il est même possible de rechercher a posteriori la décision optimale à prendre dans un état relativement rare, en le prenant systématiquement comme état initial.

Cas d'application

Le cas choisi

Le cas d'application choisi concerne une constellation homogène de 32 satellites en orbite polaire dont la mise à poste s'effectue via une orbite de dérive (figure 5 et 6) :

- Les 32 satellites sont répartis sur 8 plans d'orbite équirépartis, et dans chaque plan les 4 satellites sont équirépartis en position sur l'orbite.
- Chaque lanceur envoie une grappe de satellites sur une orbite qui dérive par rapport aux orbites finales puis chaque satellite s'injecte de manière autonome sur son orbite propre.
- Les orbites intermédiaires dérivent d'un plan par mois par rapport aux orbites finales
- Les lanceurs utilisés ont une capacité de 2, 4 ou 6 satellites.
- Le coût, la fiabilité et la durée de mise à disponibilité des lanceurs varient suivant leur capacité.
- Les satellites sont identiques avec un taux de panne constant et une même durée de vie (8 ans).
- Le coût de l'indisponibilité du service rendu par la constellation dépend de sa dégradation (nombre de satellites opérationnels manquants)

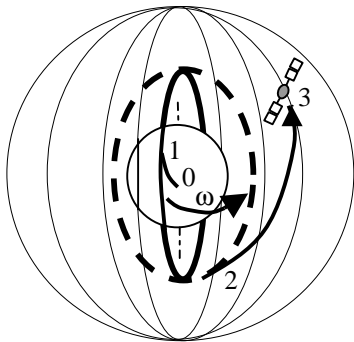


Figure 5 - mise à poste via une orbite de dérive

Chaque lanceur éjecte plusieurs satellites sur une orbite d'altitude inférieure aux orbites opérationnelles (0→1), qui dérive alors par rapport à celles-ci (1→2). La mise à poste définitive s'effectue par chacun des satellites de manière autonome quand il se situe juste en dessous de son orbite finale (2→3).

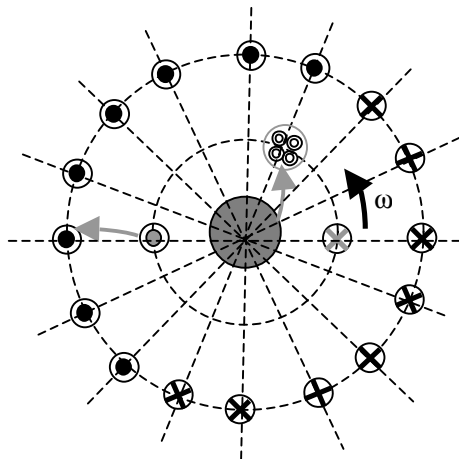


Figure 6 – Constellation de satellites vue de dessus

Mise en œuvre

La résolution d'un tel problème nécessite de procéder suivant plusieurs étapes successives.

Formalisation du problème Il convient, en premier lieu, de modéliser le système étudié en adoptant une représentation synthétique des états et des actions, puis de définir sa dynamique propre. Il est, en effet, impossible d'énumérer tous les états de la constellation et les diverses 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 présente ainsi sous forme d'une liste de la manière suivante :

$e = (e^{op}, e^{sp}, e^{sol})$ où
 $e^{op} = (x_1, x_2, \dots, x_{32})$ avec x_i la durée de vie restant avant la fin de vie du satellite opérationnel i ($x_i = 0$ si i est en panne)
 e^{sp} une liste similaire définissant les satellites de rechange restant 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 :

$a = (a^{sol}, a^{lanc}, a^{sp})$ où
 a^{sol} une liste de décisions de tir de satellites
 a^{lanc} une liste de choix d'orbites de dérive
 a^{sp} une liste de transferts de satellites d'orbites de dérive vers des orbites opérationnelles

Réalisation du simulateur Cette étape n'a posé aucun problème particulier et a pu être aisément validée.

Algorithme d'apprentissage La combinaison de trois types d'actions élémentaires dans ce problème (a^{sol} , a^{lanc} , a^{sp}) conduit à un nombre considérable d'actions différentes réalisables chaque mois (correspondant au pas de simulation choisi), ce qui est très préjudiciable à la rapidité du traitement. La solution adoptée pour répondre à cette difficulté a été de considérer les actions élémentaires comme indépendantes et séquentielles. La politique courante a pu alors être définie à partir de 3 fonctions de valeur, propres à chaque type d'actions. Compte tenu du nombre considérable d'états différents du système, ces fonctions ont été "approchées" par des combinaisons linéaires entre les paramètres suivants :

- Age des satellites opérationnels
- Nombre de satellites de rechange par orbite
- Age moyen des satellites de rechange par orbite
- Age du plus jeune satellite de rechange par orbite
- Age du plus vieux satellite de rechange par orbite
- Nombre de satellites prévus par mois dans la chaîne de lancement

Algorithme et validation L'architecture générale de l'algorithme est présentée à la figure 7. Outre le simulateur et le stockage de la fonction de valeur celui-ci comprend un algorithme d'apprentissage de la fonction de valeur **pour la politique courante** ainsi qu'un algorithme d'amélioration de cette politique. Il faut, en effet, noter que la valeur attribuée aux états du système, à partir des résultats d'un simulateur, ne correspond plus aux coûts cumulés relatifs à la politique optimale, mais à la politique réellement suivie. Il apparaît souhaitable de valider séparément ces deux algorithmes. C'est pourquoi on procède généralement en deux étapes. La première consiste à apprendre une politique prédéfinie, dite experte, tout en inhibant l'algorithme d'amélioration de la politique. Pour s'assurer de la convergence de la politique apprise vers la politique experte, on calcule l'erreur quadratique entre la fonction de valeur apprise et les coûts cumulés générés par le simulateur durant N pas successifs de simulations (fonction de valeur "approchée" de k paramètres dans la formule [6]).

$$\sqrt{\frac{\sum_{n=1}^N (C_n - V[\omega_1(e_n), \dots, \omega_k(e_n)])^2}{N}} \quad [6]$$

Après avoir vérifié que cette erreur se stabilisait au bout d'un certain temps à une faible valeur, la deuxième étape consiste à activer l'algorithme d'amélioration de la politique et vérifier l'efficacité de celui-ci ; la politique adoptée devant s'améliorer par rapport à la politique experte en conduisant à une diminution des coûts cumulés sur une certaine durée.

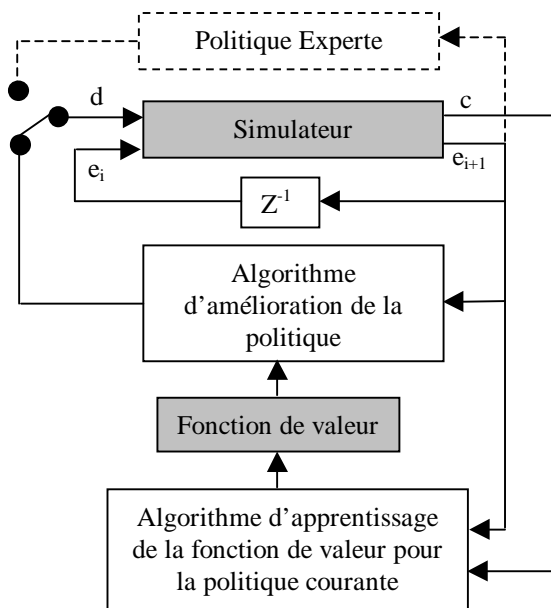


Figure 7 – Algorithme général

Résultats

Les premiers résultats obtenus sur ce cas d'application ne se sont pas révélés à ce jour totalement satisfaisant. En effet, si la première étape de validation de l'algorithme d'apprentissage a été menée à son terme, la politique suivie n'a pu être améliorée.

Ce constat peut avoir pour cause :

- Une erreur de codage informatique (peu probable)
- Une modélisation trop simpliste des actions (séparation en trois type d'actions élémentaires indépendantes et séquentielles)
- Une modélisation trop simpliste des fonctions de valeur (modèle linéaire)
- Un choix de paramètres non représentatif des états du système
- Une inadaptation de la méthode utilisée à un problème d'une telle complexité (l'effet d'un lancement de satellites de rechange peut se manifester très tardivement sur l'état de la constellation).

Conclusions

La mise en œuvre de la Programmation Dynamique avec Apprentissage par Renforcement sur un problème aussi complexe que celui qui a été choisi dans cette étude est évidemment très délicate.

Elle devait permettre de démontrer l'efficacité de cette méthode sur un cas réel très dimensionnant et lui conférer ainsi un statut privilégié pour traiter les problèmes de maintenance.

Une première tentative de résolution du problème n'a pas donné les résultats escomptés mais sera poursuivie prochainement. Un cas d'application éventuellement simplifié pourrait être alors envisagé.

La Programmation Dynamique avec Apprentissage par Renforcement reste cependant à nos yeux une méthode très prometteuse pour résoudre les problèmes de maintenance de systèmes complexes et répondre à des enjeux considérables dans des domaines variés (aéronautique, armement, espace, nucléaire, productive....).

Mots-Clés

Sûreté de Fonctionnement - Fiabilité - Disponibilité - Optimisation - Intelligence Artificielle - Constellation de satellites - Programmation Dynamique - Apprentissage par Renforcement.

Références

[1] R. S. Sutton , A. G. Barto, *Reinforcement Learning : An Introduction*, MIT Press, Cambridge, MA, 1998.

[2] F. Garcia, *Rapport d'avancement : Optimisation de systèmes complexes en Sûreté de*

Fonctionnement INRA Rapport UBIA N° 1998/4 déc 98.

[3] 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.

[4] A. Cabarbaye, J. Séroi - *Optimisation dans le domaine de la Sécurité de Fonctionnement* - Congrès $\lambda\mu$ 12, 28-30 mars 2000 Montpellier.

[5] A.Cabarbaye - Julien Séroi - L. Tomasini, *Optimisation de la Sécurité de Fonctionnement des systèmes spatiaux*, 3^e Congrès International de Génie Industriel, Montréal 26 - 28 mai 1999.

[6] A. Cabarbaye, *Modélisation et évaluation des systèmes* - Cours de technologies spatiales, Edition Cepadues Toulouse 1998.