Projet de Fin d'Études

Spécialité : Recherche opérationnelle, optimisation
Année scolaire : 2022-2023

Méthodes de résolution de problèmes de tournées multi-véhicules avec fenêtres de temps et coûts dépendants du temps pour la gestion de constellation de satellites

Mention de confidentialité :
Rapport non confidentiel et publiable sur internet

Auteur : Romain BARRAULT **Promotion : 2023**

Tuteur ENSTA Paris : Zacharie ALÈS
Tuteurs organisme d'accueil : Gauthier PICARD, Cédric PRALET

Stage effectué du 11/04/2023 au 29/09/2023

Nom de l'organisme d'accueil : ONERA, centre de Toulouse
Adresse : 2 avenue Edouard Belin, Toulouse 31000

Note de confidentialité :

Ce document n'est pas confidentiel. Il peut être communiqué à l'extérieur sous format papier et diffusé sous format électronique.

Table des matières

1 Introduction

1.1 Contexte

Mon stage s'est réalisé au sein de l'ONERA Toulouse. Il s'inscrit dans le cadre du projet LICHIE (Lion Chaîne Image Élargie), coordonné par Airbus Defence and Space. Ce projet a pour but de prévoir la conception de constellations de satellites d'observation terrestre (abrégé *EOS* pour *Earth Observation Satellite*) et leur gestion. Ces satellites artificiels en orbite terrestre sont utilisés pour effectuer des prises de vue, depuis l'espace, de lieux précis sur Terre pour des usagers qui en auraient fait la demande ; ils sont notamment utilisés à des fins météorologiques, de gestion des risques naturels, ou liées de défense. Or il est important d'effectuer des prises de vue aussi qualitatives que possible, soit, notamment, lorsque le satellite est au plus proche du zénith du point de vue du site à observer. Cela ne pose pas de problème pour une demande isolée, mais lorsque la constellation de satellites doit réaliser un grand nombre de prises de vue durant une journée, cela entraîne une grande combinatoire : dans le pire des cas, il faut sélectionner de prime abord les demandes que l'on pourra effectivement traiter, et les attribuer une par une spécifiquement à un satellite de la constellation, ordonner les demandes pour chaque satellite, puis enfin décider de l'instant précis de prise de vue pour chaque demande. Tout ceci doit se faire dans le but d'aboutir à la meilleure "qualité globale" pour l'ensemble des prises de vue (nécessitant donc l'élaboration d'un critère de qualité), en sachant de plus que tout satellite est soumis à des temps de transitions entre deux prises de vue, caractérisant la réorientation de son instrument d'observation nécessaire pour passer d'un pointage vers un site à un pointage vers un autre site.

1.2 Analogie avec le *TD-TOPTW with time-dependent profits*

On peut alors établir une analogie naturelle entre ce problème et le *Team Orienteering Problem*, abrégé *TOP*. Il est caractérisé par un groupe d'individus pouvant visiter un ensemble de lieux, auxquels sont attribués individuellement un score et un temps de service. On dispose également du temps de trajet/transition d'un lieu à un autre pour tout couple de lieux. Le but est alors de maximiser le score total obtenu en visitant les lieux sous une contrainte limitant la durée totale de la visite. Ce problème est analogue au problème posé en section 1.1 pour lequel :

- les individus réalisant les visites sont les EOS ;
- les lieux à visiter sont les sites pour lesquels il existe une demande de prise de vue (qu'on appellera également *acquisition*) ;
- les temps de transition d'un lieu à un autre correspondent aux durées de réorientation de l'instrument d'observation de l'EOS ;
- le temps de service d'un lieu correspond à la durée de la prise de vue ;
- le score d'un lieu quantifie la récompense associée à la prise de vue d'un site.

Ainsi, par abus de langage, on confondra les termes *visite* et *prise de vue*, et les termes *temps de transition* et *durée de réorientation de l'instrument*. Par ailleurs, on choisit d'intégrer les temps de services des sites au sein des temps de transition d'un site à un

autre : on ne manipulera donc plus cette notion de temps de service désormais.

Il est également à noter que le problème posé en section 1.1 est notoirement plus contraint que le *TOP*. En effet, tout satellite ne peut pas visiter tout site à tout instant : il suffit que ce dernier soit situé sur la face cachée de la Terre depuis le point de vue du satellite pour qu'il soit impossible à visiter. De plus, même si ce site est visible physiquement par un satellite, il n'est peut-être pas judicieux de le visiter lorsque l'angle d'observation est trop grand, dans la mesure où cela dégraderait trop la qualité de la prise de vue. Ainsi, on considère, pour chaque couple satellite-site, des fenêtres temporelles au sein desquelles la prise de vue peut être effectuée pour être admissible : le problème devient alors un *TOP with Time Windows* (noté *TOPTW*), dans lequel un individu ne peut visiter un lieu qu'au sein d'une fenêtre temporelle propre à l'individu et au lieu.

Enfin, il reste deux nuances à apporter :

— le temps de transition entre la visite de deux sites consécutifs dépend de la position actuelle du satellite, elle-même dépendante du temps ;

— le score associé à la visite d'un site doit prendre en compte l'angle d'observation du site (cf. figure 1) : plus celui-ci est petit, plus la prise de vue est qualitative. Le score associé à la visite d'un site dépend donc également du temps.

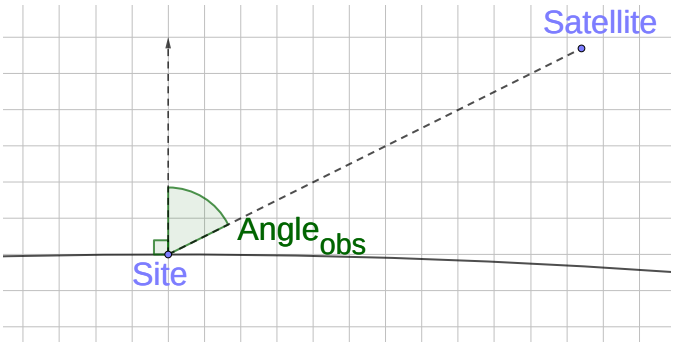

FIGURE 1 – Illustration de l'angle d'observation en $2D$

Par conséquent, par rapport aux problèmes existants en recherche opérationnelle, le problème rigoureusement analogue au problème posé en section 1.1 est le *Time-Dependent Team Orienteering Problem with Time Windows and Time-Dependent Profits*, noté *TD-TOPTW-TDP*.

1.3 Approche retenue

L'ensemble des aspects à considérer dans le problème en font un problème hautement combinatoire, au point où il est difficilement concevable de s'appuyer uniquement sur un modèle MILP pour le résoudre. C'est pourquoi une méthode de résolution reposant sur des traitements itératifs de sous-problèmes à l'aide de métaheuristiques a été imaginée. Dans la démarche globale, le problème a été découpé en trois niveaux de complexité afin de mieux comprendre les enjeux en partant des configurations les plus simples, et dans l'espoir de réutiliser à chaque fois des méthodes de résolution des niveaux inférieurs :

1. **Problème de fixation des dates** : dans ce problème simplifié, un unique satellite doit visiter tous les sites d'un ensemble donné, dans un ordre donné. La décision concerne alors uniquement les dates des visites.
2. **Problème d'ordonnancement** : dans ce problème simplifié, un unique satellite doit visiter certains sites d'un ensemble donné, mais l'ordre et les dates de visite sont tous deux à déterminer, en plus de l'ensemble effectif de prises de vue à réaliser.
3. ***TD-TOPTW-TDP*** : on dispose dans ce cas d'une constellation de satellites dans son intégralité. On doit sélectionner un sous-ensemble de prises de vue à traiter, les attribuer à un unique satellite, puis donner un ordre et fixer les dates de la séquence de prises de vue pour chaque satellite.

Les dépendances entre ces trois problèmes apparaissent plutôt clairement : résoudre le problème de fixation des dates permet d'évaluer la qualité d'une séquence fixée de prises de vue pour le problème d'ordonnancement, tandis que le problème d'ordonnancement peut être considéré comme un sous-problème du *TD-TOPTW-TDP* dans la mesure où il évalue la pertinence d'un ensemble de prises de vue attribué, sans ordre préalable, à un satellite.

Comme indiqué précédemment, ce dernier problème est hautement combinatoire, et pour des besoins industriels, la durée dédiée à sa résolution doit être de l'ordre de la quinzaine de minutes. Une durée aussi courte permet de construire le plan de prises de vue avec les informations météorologiques les plus récentes possibles (même si elles ne sont pas prises en compte au cours de cette étude) et avec une connaissance la plus à jour possible d'éventuelles prises de vue à retenter. Concrètement, on attend au maximum avant le passage d'un satellite au-dessus du centre de contrôle. Une attention toute particulière doit donc être portée au rapport qualité de la solution/temps de calcul pour les sous-problèmes. Pour ce stage, nous avons abordé les deux premiers sous-problèmes, à savoir les problèmes de fixation des dates et d'ordonnancement. Les méthodes développées dans ce contexte sont présentées dans la suite de ce rapport.

2 Revue de la littérature

De nombreux articles de la littérature traitent de la gestion de tournées de véhicules ou de la planification pour des satellites, au travers de divers problèmes d'optimisation mathématique.

Dans [1], les auteurs exhibent, pour un problème de tournées de véhicules avec routes multiples, un algorithme d'*Adaptative Large Neighbourhood Search*. Le problème traité, intitulé *VRPMTW* (pour *Vehicle Routing Problem with Multiple routes and Time Windows*) est caractérisé par un graphe orienté où les sommets représentent un ensemble d'usagers, auxquels on associe un gain, un temps de service et une fenêtre temporelle au sein de laquelle l'usager peut être desservi, et où chaque arc (i, j) est valué par la distance et/ou le temps de trajet entre les usagers i et j. On dispose également d'une flotte de taille fixe de véhicules destinés à desservir tous ces usagers ; le but est alors, dans un ordre lexicographique, de maximiser le nombre d'usagers desservis, puis de minimiser la distance totale parcourue par la flotte de véhicules. L'algorithme présenté dans cet article applique localement des opérateurs de destruction et de reconstruction à trois niveaux différents de la solution courante : à l'échelle de l'ensemble des usagers d'une route d'un véhicule, à l'échelle de l'ensemble des routes suivies par un véhicule au sein de la journée, et à l'échelle de l'ensemble des itinéraires des véhicules au cours de la journée. La sélection (aléatoire) des opérateurs est également pondérée en fonction de leur efficacité passée au cours de la résolution.

La programmation par contraintes a également été considérée pour résoudre des problèmes d'allocation d'orbites ou de planification d'observations [2, 6, 7]. Les deux premiers articles traitent le problème d'allocation d'orbite pour des usagers qui en auraient fait la demande [2, 6], et contrairement au problème traité au cours de ce stage, la planification des observations ne fait pas partie du problème puisqu'elle est alors déléguée aux usagers ayant obtenu les portions d'orbites. Dans ce problème, on appelle *mode* une possibilité d'allocation satisfaisant une certaine requête. Dans [2] les auteurs présentent un algorithme composé de trois modules : le premier génère des modes candidats pour toute requête ; le second a pour but de sélectionner les modes optimaux au regard d'un critère utilitariste ou leximin, en parcourant l'espace des solutions à l'aide d'un solveur de programmation par contraintes ; le dernier détermine la meilleure allocation de créneaux au regard de l'angle d'observation des satellites ou de la couverture nuageuse. Dans [6], l'auteur traite le *Earth Observation Satellite Constellation Scheduling Problem* (noté *EOSCSP*), caractérisé par un ensemble $\mathcal{S}$ de satellites en orbite terrestre, un ensemble $\mathcal{U}$ d'usagers, un ensemble $\mathcal{R}$ de requêtes et un ensemble $\mathcal{O}$ d'observations à planifier pour satisfaire les requêtes de $\mathcal{R}$; les usagers sont également restreints à des fenêtres temporelles exclusives au sein desquelles leurs observations ont nécessairement lieu. Il exhibe plusieurs approches par enchères, qui se distinguent légèrement au niveau des opérateurs utilisés et du moment où ils sont utilisés. Ici, les enchérisseurs sont les usagers exclusifs, et les biens sont les requêtes non exclusives émises par un centre de mission jouant le rôle de commissaire-priseur ; un deuxième algorithme utilise de la programmation par contraintes distribuée, où les variables appartiennent à des agents, représentant

l'ensemble des usagers exclusifs. Ce dernier algorithme ainsi qu'un des algorithmes par enchères, s'avèrent les plus efficaces sur les instances les plus conflictuelles, mais ils ne se comportent pas notoirement mieux qu'un algorithme glouton sur les instances réalistes.

Dans [7] les auteurs traitent la planification d'observations pour une constellation d'*EOS* en prenant en compte des demandes d'observations plus complexes : monoscopiques (réalisation d'une seule prise de vue, tout comme les demandes de notre problème), stéréoscopiques (nécessitant deux photos avec des angles de prise de vue différents), périodiques (par exemple 4 fois par jour) et systématiques (à chaque passage d'un satellite au-dessus du polygone à couvrir). Ils utilisent également la notion de mode présentée plus tôt. L'objectif consiste alors à déterminer un ensemble de requêtes, un mode par requête et une séquence d'observations permettant d'aboutir à la meilleure récompense globale. Un premier algorithme présenté utilise un solveur de programmation par contraintes, et d'autres ont recours à une décomposition du problème et à une décomposition du type *Large Neihbourhood Search* (LNS).

Les articles se rapprochant le plus de notre problème sont [8], [4] et [5]. Dans [8] les auteurs traitent le *TOPTW-TDP* discret, c'est-à-dire le même problème avec des temps de transition indépendant du temps et une discrétisation du temps. Ils résolvent ce problème à l'aide d'un algorithme de colonie d'abeilles artificielle hybridée avec du recuit simulé pour s'extraire des optima locaux. Cet algorithme trouve de meilleures solutions en un temps réduit comparé à une approche MILP (*Mixed Integer Linear Programming*) s'appuyant sur l'outil CPLEX et à un algorithme de recuit simulé rapide. Dans [4], les auteurs abordent, dans le cadre de planification pour un satellite d'observation, le *TD-OP-TDP*, soit *Time-Dependent Orienteering Problem with Time-Dependent Profits*, c'est-à-dire le même problème que nous traitons dans ce manuscrit mais en se restreignant à un seul satellite et en laissant la possibilité d'effectuer des prises de vue au cours de plusieurs orbites. Ce problème est résolu à l'aide d'un algorithme de programmation dynamique bidirectionnel combiné avec une recherche locale itérative, permettant d'insérer des visites à un endroit optimal dans la liste de visites. Dans [5], les auteurs exhibent un algorithme exact pour la planification d'activités d'EOS avec profits dépendants du temps, soit le problème traité dans ce rapport avec un unique satellite et des temps de transition indépendants du temps. Cet algorithme a recours à de la programmation dynamique à direction adaptative avec relaxation de l'espace d'états décrémentale, qui permet temporairement d'obtenir des chemins non élémentaires dans la solution.

Enfin, l'article [3] offre une perspective intéressante sur les critères de préférence globale pour les solutions à des problèmes temporels simples. Un tel problème est défini par un couple $(\mathcal{V}, \mathcal{C})$ avec $\mathcal{V}$ un ensemble de variables v_i représentant des instants, et $\mathcal{C}$ ensemble de contraintes souples, définies comme un couple $\langle I, f_{ij} \rangle$ où I est un intervalle et f_{ij} une fonction de préférence locale sur I qui associe à la valeur de $v_j - v_i$ un score dans un ensemble totalement ordonné. Trois types de préférences globales sont introduits : maillon faible, Pareto et utilitariste. Cet article exhibe également, pour chacun des types de préférences globales, un algorithme permettant d'aboutir à l'optimum en temps polynomial.

Finalement, par rapport aux travaux existants décrits précédemment :

— [3] permet de remettre en question les critères qu'on pourrait considérer comme "classiques" pour des modèles de programmation linéaire ou des heuristiques d'optimisation. Il a permis d'inspirer le critère de préférence (en l'occurrence, leximin) pour la conception d'une méthode de résolution (cf. section 3.3.2) ;
— [8], [4] et [5] peuvent servir de comparatifs afin d'estimer les performances de l'algorithme conçu pour résoudre le problème posé, tant les problèmes qu'ils traitent sont semblables au nôtre.
— Les autres articles peuvent servir d'inspiration pour établir la structure de l'algorithme de résolution, mais les problèmes qu'ils abordent s'éloignent légèrement du *TD-TOPTW-TDP*.

Nous détaillons maintenant les méthodes que nous avons développées pour les problèmes de fixation des dates et d'ordonnancement.

3 Traitement du problème de fixation des dates

Dans cette partie, nous le sous-problème le plus élémentaire du *TD-TOPTW-TDP*, ainsi que les résultats obtenus.

3.1 Données d'entrée

Les paramètres du problème de fixation des dates sont présentés ci-dessous :

1. un ensemble de sites $N = \{1, 2...n\}$ numérotés de 1 à n, à visiter *dans l'ordre* $1, 2...n$ pour un satellite en orbite terrestre ;
2. $\forall i \in N$, une fenêtre temporelle $[a_i, b_i]$ (noté TW pour *Time Window*) de possibilité de visite du site i ;
3. $\forall i \in [\![1; n-1]\!]$, une fonction de temps de transition entre les sites successifs i et $i+1$ notée
   $\Delta_i : t \to \Delta_i(t)$, linéaire par morceaux et vérifiant de plus $t \to t + \Delta_i(t)$ croissante (plus une transition commence tôt, plus elle finit tôt) ;
4. $\forall i \in N$, une fonction de récompense $r_i : t \to r_i(t)$ de visite du site i à l'instant t, également linéaire par morceaux.

Le fait que les fonctions de récompense et de temps de transition soient supposées linéaires par morceaux est une propriété essentielle pour un des trois modèles présentés plus bas ; en particulier, on aura recours, pour $i \in N$, à une subdivision de $[a_i, b_i]$ en K segments (nombre potentiellement variable), subdivision notée :

$$\sigma_i = (\sigma_i^0 = a_i, \sigma_i^1, ...\sigma_i^{K-1}, \sigma_i^K = b_i)$$

On note alors, pour $i \in N$ et $k \in [\![1; K]\!]$, r_i^k et Δ_i^k respectivement les interpolations linéaires de r_i et Δ_i entre σ_i^{k-1} et σ_i^k.

3.2 Présentation des instances

Pour ce problème, les instances sont générées de manière assez triviale. Le satellite (en orbite basse) effectue un trajet du pôle Sud au pôle Nord à une certaine altitude, et n acquisitions sont placées aléatoirement sur son chemin. Dans le but d'aboutir à un ordre de visite pertinent, ces acquisitions sont placées dans l'hémisphère Nord avec une latitude croissante ; un paramètre Δ_{lat}^{max} délimite la différence de latitude entre deux acquisitions successives. La longitude des acquisitions est définie aléatoirement dans l'intervalle $[-\Delta_{long}^{max}; \Delta_{long}^{max}]$, ce dernier paramètre étant fixé au préalable de manière à ce que les cibles considérées au sol soient effectivement survolées par le satellite au cours de son passage.

La figure 2 représente un exemple d'instance pour ce problème : le maillage bleu représente la surface de la Terre, les cibles des acquisitions sont représentées par les cercles verts, et les valeurs utilisées pour les paramètres sont : $\Delta_{lat}^{max} = 3.1^o$ et $\Delta_{long}^{max} = 1.5^o$.

Leurs TW d'acquisition sont définies par les TW au sein desquelles le satellite a un angle d'observation de la cible concernée (cf figure 1) inférieur à une valeur seuil α_i^{max},

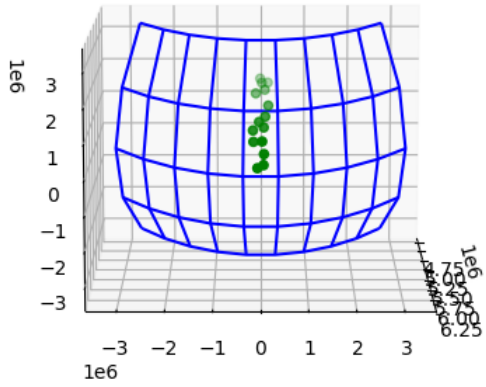

FIGURE 2 – Exemple d'instance pour le problème de fixation des dates

égale à 40 ou 60^o. On dispose également d'une fonction qui, à partir d'un instant t et d'un site i, renvoie l'angle d'observation du satellite pour le site i à l'instant t ; de même pour les temps de transition (pour $i < n$). Or, si cette première fonction n'est déjà pas triviale, la seconde n'est même pas analytique : cela pose un problème pour considérer le temps comme une variable continue. Pour pallier ce problème, on effectuera des calculs de ces grandeurs à des instants précis (ce qui définira la subdivision), et on approximera leurs valeurs réelles par une interpolation linéaire, pour aboutir aux fonctions Δ_i et r_i souhaitées. Deux exemples sont donnés en figures 3 et 4.

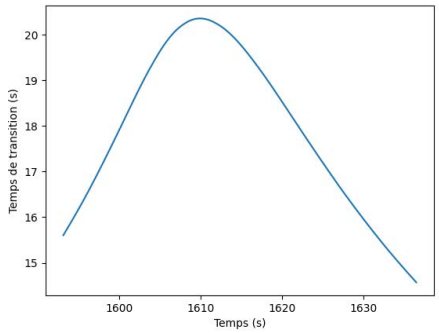

FIGURE 3 – Exemple de fonction de temps de transition en fonction du temps

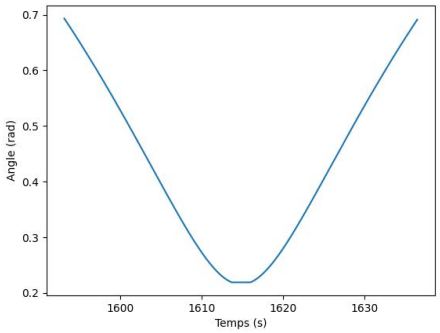

FIGURE 4 – Exemple de fonction d'angle d'observation en fonction du temps (avec $\alpha_i^{prox} = 0.22$rad et $t_i^{\text{prox}} = 1613$s)

Enfin, pour chaque acquisition $i \in N$, on définit un angle α_i^{prox} comme étant le meilleur angle d'observation du satellite pour le site i parmi 300 mesures d'angle réparties uniformément au sein de la TW $[a_i, b_i]$ (atteint à un instant noté t_i^{prox} cf figure 4). C'est à l'aide de cette valeur que l'on définit la fonction de récompense r_i.

Plus précisément, deux formes de fonction de récompense ont été imaginées : l'une servant un critère *utilitariste* (notée $r_i^u(t)$), l'autre servant un critère *égalitariste* (notée $r_i^e(t)$).

Le *critère de récompense utilitariste* est défini par :

$$r_i^u(t) = 1 - \frac{0.9(\alpha_i(t) - \alpha_i^{\text{prox}})}{\alpha_i^{\max} - \alpha_i^{\text{prox}}}$$

La quantité r_i^u correspond ainsi à une fonction linéaire qui renvoie 1 à la date t_i^{prox} et la valeur 0.1 à la pire date d'observation possible.

Le *critère de récompense égalitariste* est quant à lui défini par :

$$r_i^e(t) = 1 - (1 - r_i^u(t))^2$$

Ce critère valorise les augmentations de récompense pour des valeurs d'angles plus grandes, contrairement au premier qui valorise toute augmentation uniformément. Ces deux critères sont représentés aux figures 5 et 6.

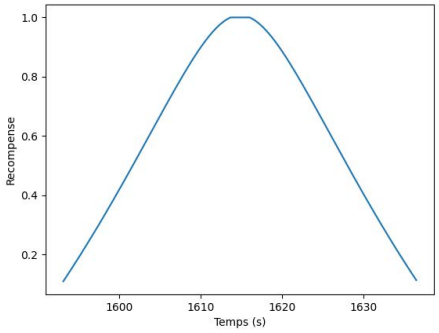

FIGURE 5 – Exemple de fonction de récompense utilitariste en fonction du temps

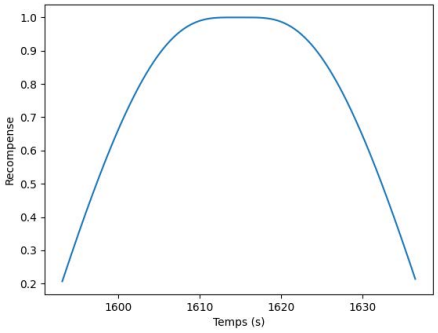

FIGURE 6 – Exemple de fonction de récompense égalitariste en fonction du temps

Par la suite, on effectuera tous les tests en utilisant le critère égalitariste dans la mesure où l'optique n'est pas de "sacrifier" certaines observations pour aboutir à d'autres observations très bonnes, mais plutôt de faire en sorte que toutes soient relativement acceptables individuellement.

Le problème et les instances étant maintenant définis, on peut introduire les trois modèles qui ont été conçus pour résoudre le plus efficacement possible le problème de fixation des dates.

3.3 Méthodes de résolution

Je présente dans cette partie les trois méthodes conçues pour résoudre le problème de fixation des dates.

3.3.1 MILP itératif

Pour cette méthode basée sur la programmation linéaire en nombres mixtes, le principe est le suivant :

1. On discrétise les fonctions d'angle d'observation et de temps de transition sur un nombre très réduit d'instants.
2. On résout le modèle MILP (défini plus bas) permettant d'aboutir aux instants optimaux de visite des sites.
3. Après résolution, on observe la différence entre d'une part les valeurs de temps de transition et d'angle obtenus via interpolation linéaire et d'autre part les valeurs réelles fournies par les fonctions de départ Δ_i et r_i. Si une de ces différences est supérieure à une précision fixée initialement, on ajoute à la discrétisation les instants problématiques et les valeurs réelles de temps de transition/angle d'observation. Cela raffine ainsi la discrétisation pour ces instants.
4. On réitère jusqu'au respect des précisions souhaitées.

L'effet d'une itération sur la fonction d'angle est représentée aux figures 7 et 8.

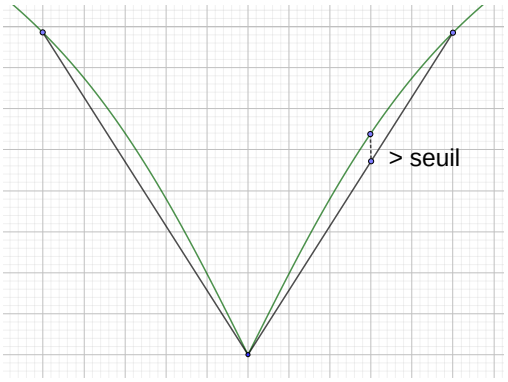

FIGURE 7 – Fonction d'angle avant itération (non-respect du seuil)

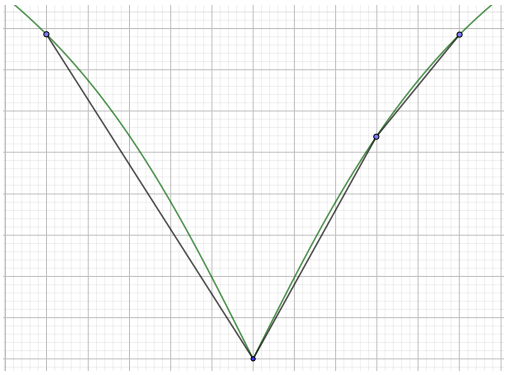

FIGURE 8 – Fonction d'angle après itération (un point de discrétisation supplémentaire)

Le modèle MILP considéré à chaque étape est donné ci-dessous :

Variables. $\begin{cases} (d_i)_{i\in N} : \text{ date de visite du site i ;} \\ (\rho_i)_{i\in N} : \text{récompense associée à la visite du site i ;} \\ (s_i^k)_{i\in N, k\in[\![1;K]\!]} : \text{ variable binaire égale à 1 à condition que } d_i \in [\sigma_i^{k-1}, \sigma_i^k] \end{cases}$

Contraintes.
$$\begin{cases} \forall i \in [\![1; n-1]\!], \forall k \in [\![1; K]\!], d_{i+1} \geq d_i + \Delta_i^k(d_i) - M(1 - s_i^k) : \text{respect des temps de transition selon l'intervalle auquel } d_i \text{ appartient (formulation big-M) ;} \\ \forall i \in N, \forall k \in [\![1; K]\!], a_i + (\sigma_i^{k-1} - a_i)s_i^k \leq d_i \leq b_i + (\sigma_i^k - b_i)s_i^k : \text{respect des TW admissibles et définition adéquate des } s_i^k ; \\ \forall i \in N, \sum_{k=1}^{K} s_i^k = 1 : d_i \text{ appartient à un unique intervalle de la subdivision ;} \\ \forall i \in N, \forall k \in [\![1; K]\!], \rho_i \leq r_i^k(d_i) + M'(1 - s_i^k) : \text{respect de la fonction de récompense sur l'intervalle auquel } d_i \text{ appartient (formulation big-M) ;} \\ \forall i \in N, d_i \in \mathbb{R}^+, \rho_i \in \mathbb{R}^+, \forall k \in [\![1; K]\!], s_i^k \in \{0, 1\}. \end{cases}$$

Objectif. $\max \sum_{i \in N} \rho_i$

Remarque. *Pour la définition des big M, il suffit de prendre* $M = \max_{i \in N, t \in \sigma_i} \Delta_i(t)$ *et* $M' = \max_{i \in N, t \in \sigma_i} r_i(t)$.

La formulation MILP risquant de passer difficilement à l'échelle (en vue d'être utilisée en tant que brique de résolution pour les sous-problèmes moins élémentaires), d'autres méthodes de résolution ont été conçues ; le but est qu'elles soient moins chronophages.

3.3.2 Réduction itérative des fenêtres temporelles

Pour cette méthode, on dispose dans un premier temps d'une fonction déterminant, à partir d'une liste d'acquisitions telle que définie en section 3.2, si le plan d'observation au plus tôt est réalisable. Le principe de cette méthode est alors de réduire itérativement les fenêtres de temps (TW) de manière à approcher l'instant du meilleur angle d'observation pour chaque visite, tant que le plan d'observation au plus tôt est réalisable (cf. figures 9 et 10).

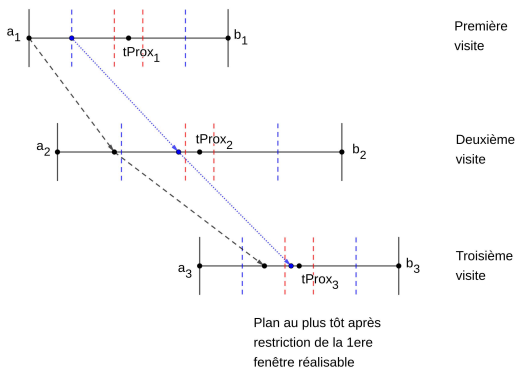

FIGURE 9 – Fenêtres temporelles avant itération

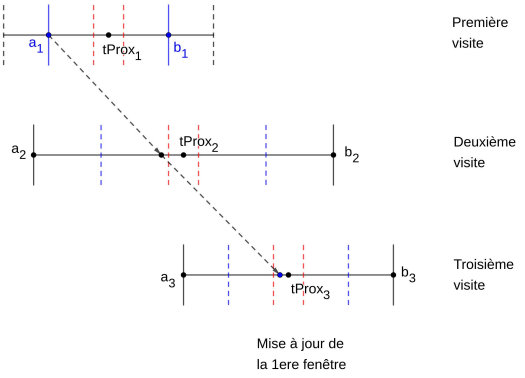

FIGURE 10 – Fenêtres temporelles après itération

Cela donne donc l'heuristique suivante :

1. On divise toutes les TW en un nombre n_{windows} de sous-fenêtres réparties uniformément.
2. On associe à chaque TW une récompense individuelle égale à $r_i(a_i)$.
3. Tant que le plan d'observation au plus tôt est réalisable, faire :
   (a) choisir la TW de récompense individuelle la plus faible, qui n'est pas déjà réduite à sa plus petite sous-fenêtre (on note i l'acquisition correspondante) ;
   (b) mettre à jour a_i et b_i après réduction de la TW ;
   (c) si le plan d'observation au plus tôt n'est pas réalisable, on annule la réduction de TW et l'acquisition i n'est plus candidate pour une réduction de TW pour la suite du processus ; on choisit alors l'acquisition dont la récompense individuelle est immédiatement plus grande que celle actuelle.

Lorsque cette boucle s'arrête, il n'est plus possible de réduire une seule TW, et on a une solution réalisable de récompense totale améliorée.

Cette méthode propose donc une approche dite « *égalitariste* », selon le critère leximin (cf. étape 3).

Remarque. *L'analyse des performances de cette méthode passera par une discussion de la valeur du nombre maximum de sous-fenêtres par fenêtre temporelle.*

3.3.3 Programmation dynamique

Le principe de la résolution pour cette méthode est le suivant :

1. On discrétise le temps selon un pas de temps T ;
2. Pour tout $i \in N$ et toute valeur $t \in TW_i = [a_i; b_i] \bigcap (T\mathbb{Z})$, on définit $r_{it} = r_i(t)$ la récompense associée à la visite du site i à l'instant t.
3. On calcule ensuite par programmation dynamique des termes R_{it} qui donnent la meilleure récompense d'un plan visitant les sites 1 à i et finissant la visite du site i à la date t. Pour cela, on utilise les opérations suivantes :
   (a) *Initialisation :* $R_{1,t} = r_{1,t}, \forall t \in TW_1$;
   (b) *Equation de programmation dynamique :*

$$\forall i \in N \backslash \{1\}, \forall t \in TW_i, R_{i,t} = r_{i,t} + max_{t' \in TW_{i-1}, t' + \Delta_{i-1}(t') \leq t} \{R_{i-1,t'}\}$$

4. La récompense totale de la solution correspond alors à $max_{t \in TW_n} R_{n,t}$, et pour obtenir cette récompense on peut fixer les dates des visites selon les argmax sur les valeurs t' utilisées lors de son calcul.

De manière plus générale, ce problème se conçoit naturellement comme un problème de plus long chemin dans un graphe orienté $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, comme représenté à la figure 11 :

— l'ensemble $\mathcal{V}$ des sommets du graphe représente l'ensemble des instants d'observation possibles pour chaque acquisition (après discrétisation du temps), partitionné en $\mathcal{V} = \bigcup_{i \in N} \mathcal{V}_i$ où $\mathcal{V}_i$ est constitué des sommets (i, t) représentants les instants possibles d'observation du site i, auquel on ajoute une source A et un puits B fictifs symbolisant le début et la fin de l'itinéraire ;

— l'ensemble $\mathcal{E}$ des arcs du graphe contient un arc d'un sommet (i,t) à un sommet $(i+1,t')$ à un sommet j si et seulement si il est possible d'effectuer la transition d'une visite du site i à la date t à une visite du site $i+1$ à la date t' au vu des contraintes temporelles ;
— tout arc est valué par la récompense individuelle liée à l'observation à la date correspondant à sa destination (valeur nulle pour un arc incident au puits).

Cette méthode est implémentée via l'algorithme décrit par le pseudo-code ci-dessous :

Algorithm 1: Programmation dynamique pour le problème de fixation des dates

Entrées : Une instance du problème de fixation des dates
Un pas de temps T pour discrétiser le temps

Initialisation :
$\forall i \in N, timer[i] \leftarrow [a_i; b_i] \bigcap T\mathbb{Z}, instantReward[i] \leftarrow [r_i(t) \text{ for } t \in timer[i]]$
$\forall i \in N \backslash \{n\}$ $globalReward[1] \leftarrow instantReward[1]$
for *i=2 ;i<n+1 ;i++* **do**
  for *k=1 ;k<len(timer[i]) ;k++* **do**
    $R \leftarrow max(globalReward[i-1])$ tel que la transition de l'argmax à $timer[i][k]$ est possible
    $ind \leftarrow indexOf(R)$ (dans $globalReward[i-1]$)
    $globalReward[i][k] = instantReward[i][k] + R$
    $indexArgmax[i][k] = ind$
  end
end
return $max(globalReward[n])$; le parcours des timers via indexArgmax permet de retrouver les instants des visites des sites.

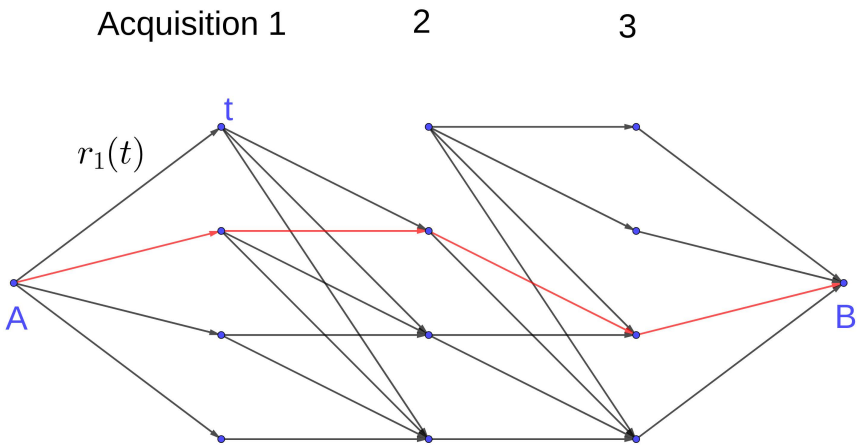

FIGURE 11 – Graphe modélisant l'algorithme de programmation dynamique décrit

Ainsi, dans l'exemple de la figure 11, il y a trois acquisitions à réaliser, et chacune

possède quatre instants d'observations admissibles. En revanche, on observe qu'il n'est pas possible d'effectuer une transition du premier instant admissible pour la première au premier instant admissible pour la deuxième, d'où l'absence d'arc entre les deux sommets correspondants. Le chemin rouge symbolise le plus long chemin de la source au puits, et les instants correspondant aux sommets parcourus décrivent les instants de visite optimaux. De plus, la récompense totale liée à cette fixation de dates est égale à la somme des valeurs des arcs empruntés.

Remarque. *L'analyse des résultats via cette méthode passera également par la variation du paramètre T.*

3.4 Résultats numériques

Les résultats des tests réalisés sur le serveur de calcul de l'ONERA sont consignés ci-après.

3.4.1 Détails des instances

Les instances sont générées aléatoirement d'après la méthode détaillée en section 3.2, et des instances différentes sont générées en modifiant la graine pour le générateur de nombres pseudo-aléatoire. L'orbite du satellite est à une altitude de $500km$ et la rotation de la Terre est prise en compte. On a également $\Delta_{lat}^{max} = 1.6^o$ et $\Delta_{long}^{max} = 3.1^o$. L'angle d'observation maximal toléré est choisi aléatoirement dans $\{40, 60\}^o$ pour chaque acquisition. Pour les tests, voici les valeurs de paramètres testées :

- Pour la méthode de réduction de TW, n_{windows} varie dans $\{10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10$
- Pour la méthode DP, T varie dans $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10\}$s.
- Pour le MILP itératif, le seuil de tolérance lié aux angles est égal à 0.005rad (pour une marge d'erreur d'environ 2.5km à la surface de la Terre) et celui lié aux temps de transition égal à 0.05s (au plus 5% du temps de transition). On commence également la résolution avec une discrétisation en 3 points pour tout $i \in N$: en a_i, en b_i et en t_i^{prox}.

À titre informatif, on fournit également des ordres de grandeur de quelques grandeurs caractéristiques du problème :

- la durée du survol du pôle Sud au pôle Nord est d'environ 20 minutes ;
- les TW ont une durée d'une cinquantaine à une centaine de secondes ;
- les transitions peuvent durer de plusieurs secondes à plusieurs dizaines de secondes.

3.4.2 Étude des paramètres

Une analyse préliminaire sur 5 instances différentes a permis d'éliminer la méthode du MILP itératif. Les résultats des tests sont consignés dans le tableau 1.

Ces résultats ont été déterminés avec des paramètres choisis relativement arbitrairement pour la programmation dynamique ($T = 1s$) ainsi que pour la réduction itérative de TW (n_{windows}) = 50. On remarque rapidement que si la récompense totale déterminée par la méthode du MILP itératif est plutôt convaincante, le temps de calcul est de

	Méthodes	MILP itératif	Programmation Dynamique	Réduction Itérative TW
Instance 1	Récompense totale	**19.58278485**	19.5571698441	18.96542976
	Temps de calcul (ms)	194	**24**	68
Instance 2	Récompense totale	19.30959300	**19.3469446022**	18.65369689
	Temps de calcul (ms)	188	**9**	55
Instance 3	Récompense totale	16.36292121	**17.4164727448**	14.87469596
	Temps de calcul (ms)	146	**8**	46
Instance 4	Récompense totale	14.18166705	**16.7753231873**	13.38955549
	Temps de calcul (ms)	315	**8**	43
Instance 5	Récompense totale	**19.03119047**	18.8949582652	17.59195936
	Temps de calcul (ms)	404	**8**	59

TABLE 1 – Résultats des différentes méthodes sur 5 instances générées aléatoirement

l'ordre de la centaine de millisecondes ; or, en vue d'être utilisée comme une sous-brique de la résolution du *TD-TOPTW-TDP*, dans l'idéal, le temps de calcul accordé à cette partie devrait être de l'ordre de la dizaine de millisecondes. Cela questionne également la pertinence des paramètres utilisés pour les deux autres méthodes. Varier les seuils de tolérance n'est pas très fructueux non plus, le solveur n'ajoutant en général pas plus de 2 points à chaque acquisition : la majeure partie du temps est prise par le solveur MILP. De plus, cette méthode présente l'inconvénient d'aboutir à des solutions potentiellement légèrement irréalisables. En effet, en raison du seuil de tolérance en valeur absolue portant sur la différence entre les temps de transition obtenus via interpolation linéaire et les temps de transition réels, ces derniers sont parfois supérieurs à ceux déterminés lors de la résolution du modèle MILP itératif. Dans ce cas, une réparation de la solution *a posteriori* est nécessaire. Au cours d'autres tests, nous avons également tenté d'imposer aux temps de transition déterminés via interpolation linéaire d'être supérieurs aux temps de transition réels, mais cela n'a pas été concluant. Il faut savoir que la forme de la fonction représentée à la figure 3 n'est pas anodine et était plutôt commune à toutes les fonctions de transition, notamment par rapport à leur concavité. Il est donc naturel qu'une interpolation linéaire d'une telle fonction aboutisse à une sous-évaluation de sa valeur réelle. Contourner ces problèmes semblait cependant présenter peu d'intérêt, notamment au vu des performances des deux autres méthodes, nous l'avons donc abandonnée dans la suite du travail.

Sur les deux méthodes restantes, nous avons effectué une exploration des paramètres :

— de pas de temps pour la méthode de programmatino dynamique ;

— de nombre de sous-fenêtres pour la méthode de réduction des TW.

Ces tests ont été effectués pour un nombre de sites $n \in \{5, 10, 15, 20\}$ et pour une vingtaine d'instances par valeur de n. Les résultats (valeur moyenne de l'objectif et temps de calcul) sont consignés ci-dessous. Les diagrammes de la ligne supérieure des prochaines figures sont dédiés à la méthode de programmation dynamique, et ceux de la ligne inférieure à la méthode de réduction des TW.

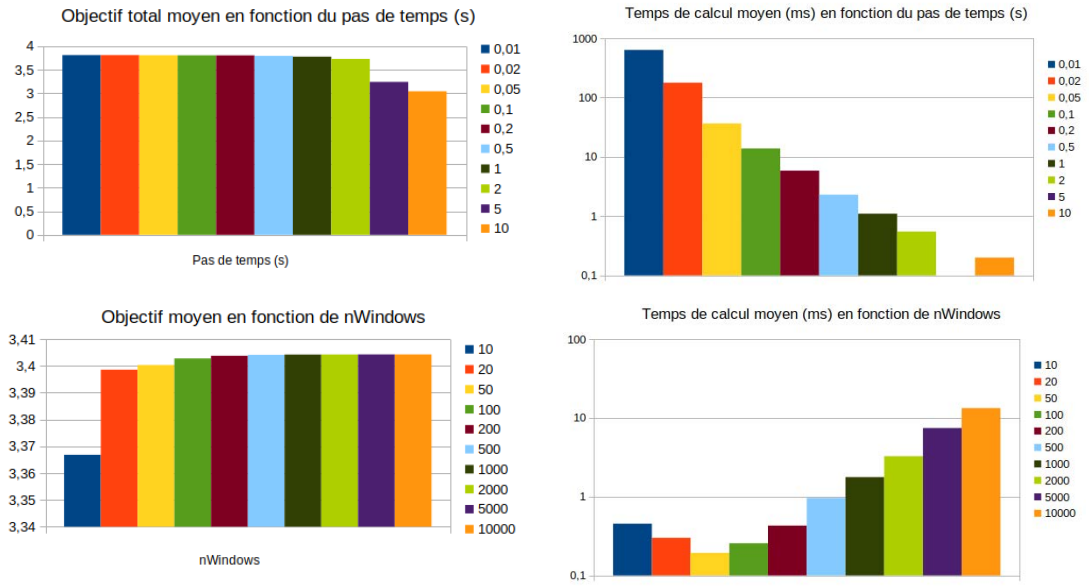

FIGURE 12 – Synthèse des résultats pour les instances de taille 5

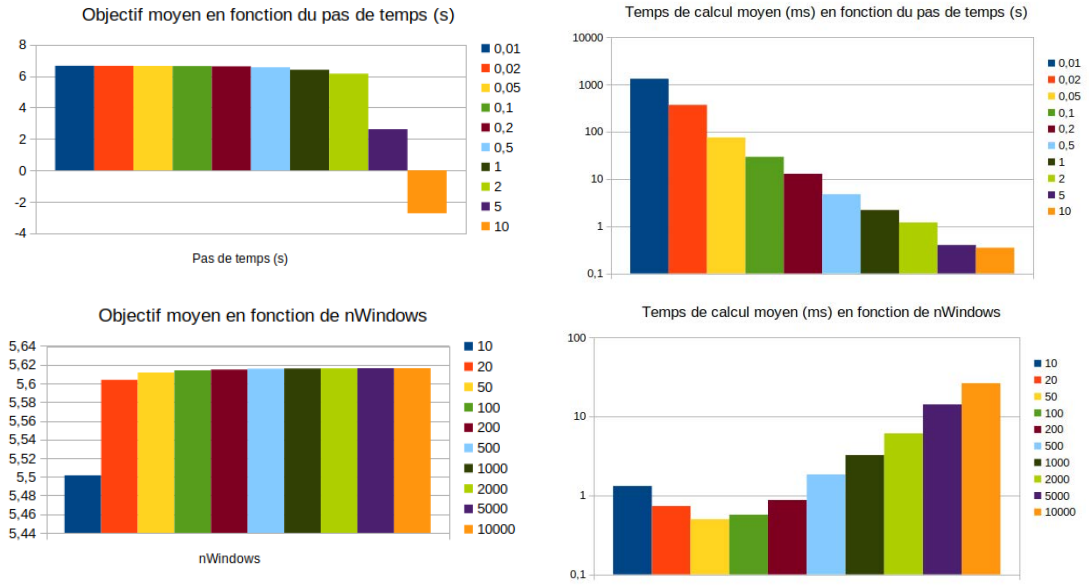

FIGURE 13 – Synthèse des résultats pour les instances de taille 10

Plusieurs remarques sont à faire : la valeur parfois négative de l'objectif dans le cas de la méthode via Programmation Dynamique (notée DP par la suite) tient du fait que sur certaines instances, le pas de discrétisation est trop grand pour aboutir à la moindre solution réalisable. Cette valeur négative tient d'une convention établie dans le code.

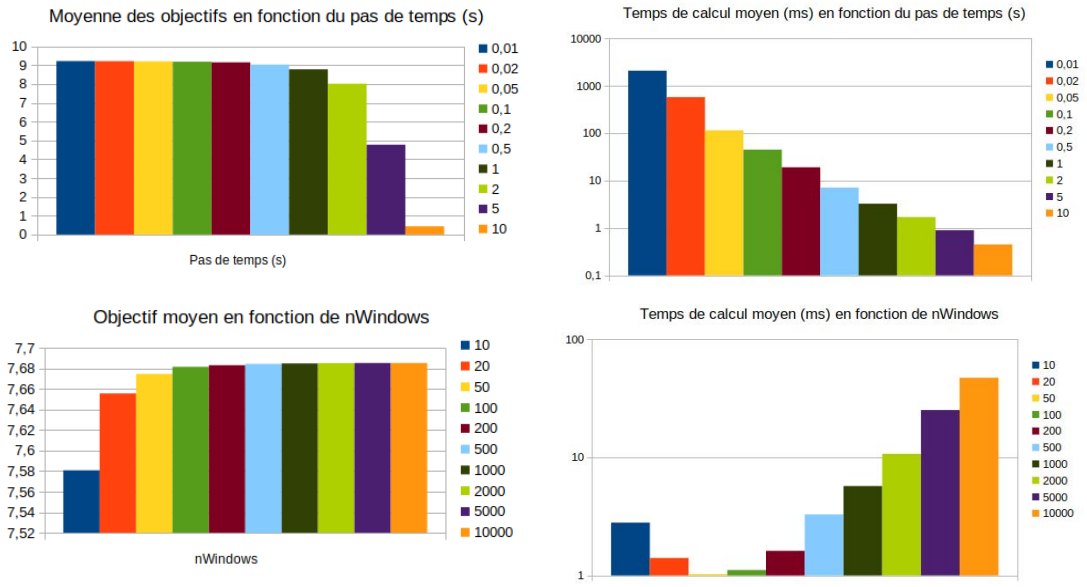

FIGURE 14 – Synthèse des résultats pour les instances de taille 15

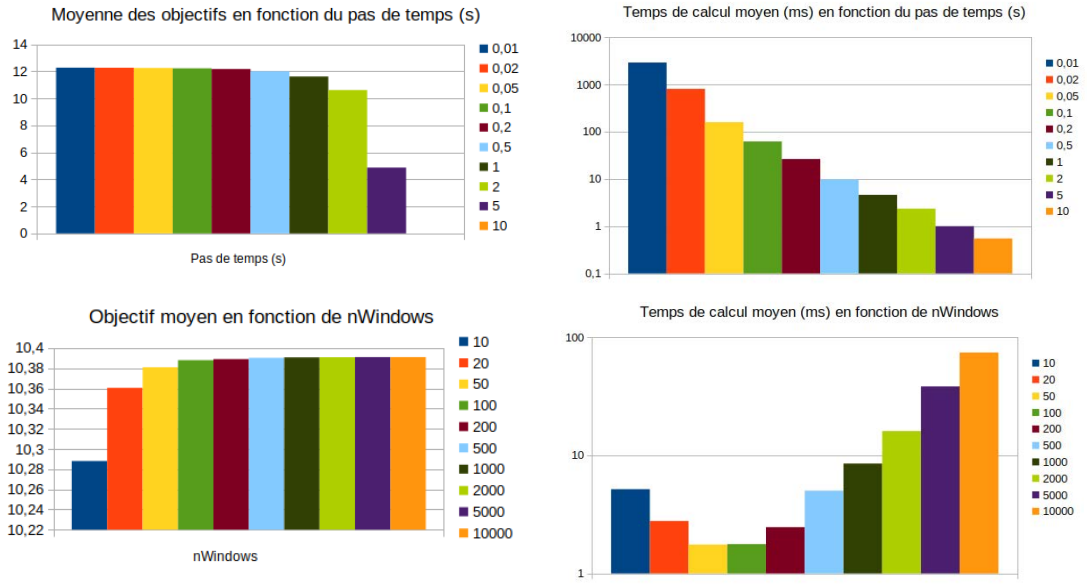

FIGURE 15 – Synthèse des résultats pour les instances de taille 20

Ces résultats soulignent l'efficacité supérieure de l'algorithme de programmation dynamique devant celui de réduction itérative des TW, quel que soit le nombre d'acquisitions de l'instance. Cette tendance sur les moyennes se vérifie également individuellement pour chaque instance. Par ailleurs, pour un bon compromis performance/temps de calcul, une

bonne valeur de T peut être sélectionnée parmi $\{0.2, 0.5, 1\}$.

Par ailleurs, les résultats ne semblent pas se démarquer notoirement par rapport à la taille des instances ; par la suite, on gardera un nombre d'acquisitions égal à 15.

4 Traitement du problème d'ordonnancement

Dans cette partie, nous le sous-problème du *TD-TOPTW-TDP* de complexité immédiatement supérieure au problème de fixation des dates, ainsi que les résultats obtenus.

4.1 Données d'entrée

Les paramètres du problème d'ordonnancement sont présentés ci-dessous :

— un ensemble de sites $N = \{1, 2...n\}$ candidats pour être visités par un satellite en orbite terrestre ;

— $\forall i \in N$, une TW $[a_i, b_i]$ de possibilité de visite du site i ;

— $\forall i, j \in N, i \neq j$, une fonction de temps de transition entre les sites i et j (pour $i, j \in N, i \neq j$) notée :
$\Delta_{i,j} : t \to \Delta_{i,j}(t)$;

— $\forall i \in N$, une fonction de récompense $r_i^e : t \to r_i^e(t)$ de visite du site i à l'instant t.

On disposera également d'une méthode de référence pour quantifier l'efficacité de la méthode développée. Cette méthode de référence est appelée l'**Heuristique only tProx**. Le principe est le suivant :

— les acquisitions sont triées en fonction de la valeur de leur t_{prox}, par ordre croissant (on espère ainsi aboutir à de bons angles d'observation pour chacune d'entre elles) ;

— on tente à chaque étape d'ajouter une acquisition supplémentaire à la fin du plan courant, si cela est temporellement faisable vu les fonctions $\Delta_{i,j}$, dans l'ordre défini au point précédent ;

— n applique la fixation des dates à l'ensemble d'acquisitions retenu.

4.2 Présentation des instances

Pour ce problème, il n'est pas adapté de procéder de la même manière que pour le problème de fixation des dates, où l'on essayait de faire en sorte que l'ordre des acquisitions soit "pertinent" pour que la fixation des dates se fasse dans les meilleures conditions. Ainsi, assez logiquement, l'heuristique only tProx aboutissait à un résultat très convaincant en un temps très restreint sur les instances de la section précédente. La nécessité de varier le type d'instances nous est alors apparue. On définit alors deux paramètres $\text{lat}_{\max}$ et $\text{long}_{\max}$, et les observations candidates sont placées en générant aléatoirement leur latitude et longitude respectivement dans $[-\text{lat}_{\max}; \text{lat}_{\max}]$ et $[-\text{long}_{\max}; \text{long}_{\max}]$. On définit alors plusieurs archétypes d'instance :

— un premier où les valeurs de ces deux paramètres sont relativement grandes, entraînant une grande aire admissible pour les lieux des acquisitions ;

— un second où ces deux valeurs sont faibles, entraînant une plus petite aire admissible mais des temps de transition supposément réduits ;

— un troisième où $\text{long}_{\max}$ est grand et pas $\text{lat}_{\max}$. Ce type d'instance semble plus difficile à gérer car davantage de sites sont survolés au même moment.
— un quatrième où $\text{lat}_{\max}$ est grand et pas $\text{long}_{\max}$. Ce type d'instance est assez semblable aux instances de la section précédente, on s'attend donc à ce que l'heuristique tProx soit performante sur ce type d'instance.

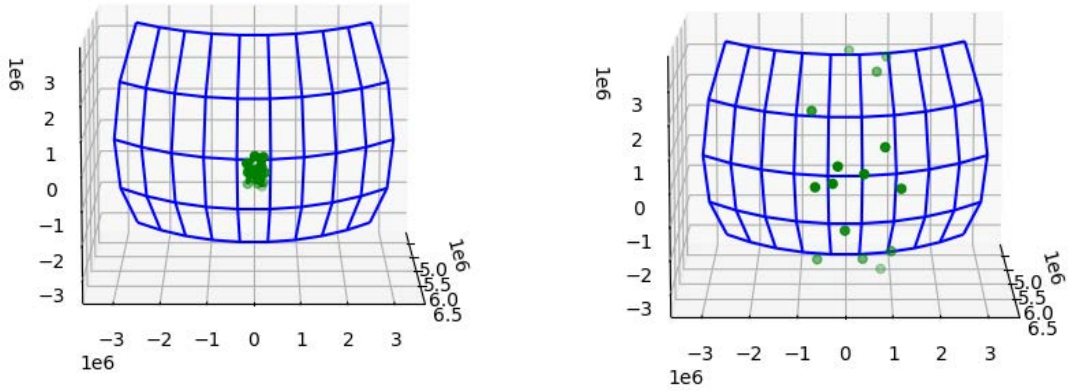

FIGURE 16 – Archétypes d'instances (1) à gauche et (2) à droite

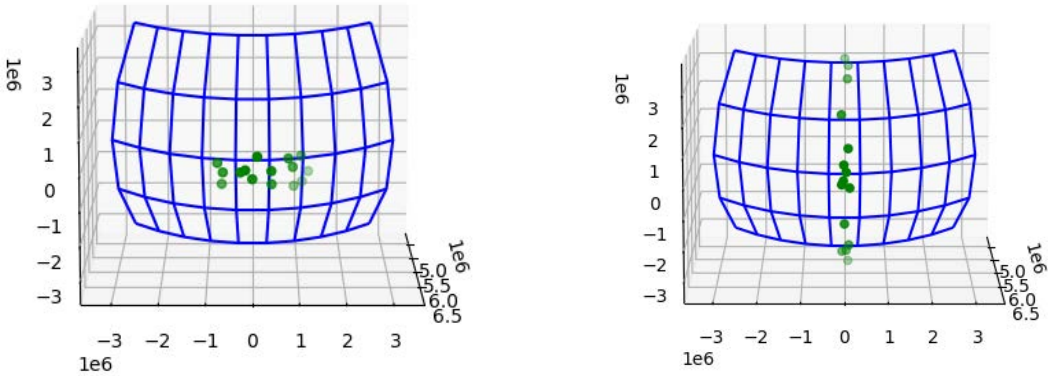

FIGURE 17 – Archétypes d'instances (3) à gauche et (4) à droite

4.3 Méthode de résolution

Pour résoudre le problème d'ordonnancement, une méthode de recherche locale aléatoire reposant sur trois opérateurs locaux a été conçue. Comme présenté ultérieurement, cette méthode met en exergue les limites de l'heuristique only tProx.

4.3.1 Génération d'une solution initiale

La première étape de la résolution est la génération d'une solution initiale. Trois méthodes ont été conçues, reposant sur une heuristique d'insertion gloutonne, où les acquisitions de l'instance sont ajoutées à la liste à traiter tant que le plan au plus tôt est réalisable, et chacune de ces méthodes repose sur un tri au préalable des acquisitions de l'instance :

— la première trie les acquisitions par ordre croissant de $tProx$, notée From tProx ;
— la seconde par ordre croissant de a_i, notée From TWS (pour Time Window Start) ;
— la dernière par ordre croissant de b_i, notée From TWE (pour Time Window End).

Il s'agit d'une manière assez simpliste d'effectuer la sélection d'acquisitions mais le but était ici de se focaliser sur l'ordonnancement des acquisitions sélectionnées, qui est en lui-même un problème complexe.

4.3.2 Opérateurs de recherche locale

Trois opérateurs locaux ont été définis pour réordonner la liste des acquisitions sélectionnées (dans les exemples ci-dessous, l'indexation commence à 0) :

— **Shift :** place une acquisition à un autre endroit dans la liste d'acquisitions, cf figure 18 ;

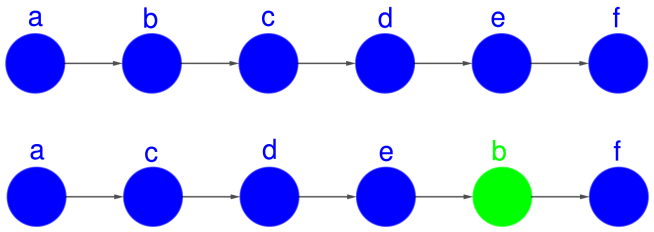

FIGURE 18 – Opérateur **Shift(b,4)**

— **Swap :** échange la place de deux acquisitions, cf figure 19 ;

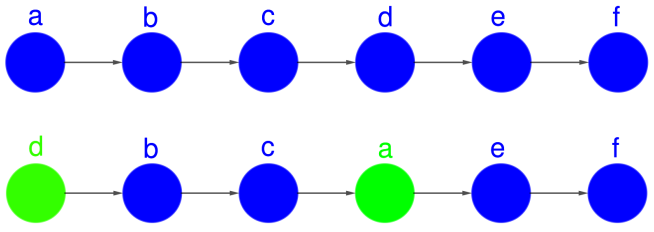

FIGURE 19 – Opérateur **Swap(a,d)**

— **Reverse :** inverse l'ordre des acquisitions entre deux indices définis, cf figure 20.

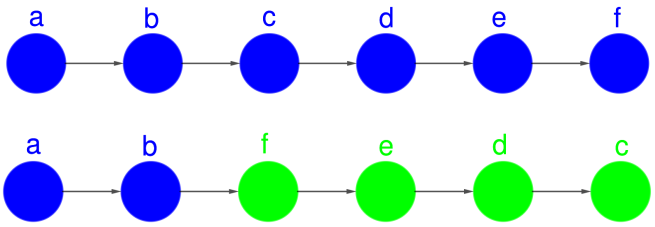

FIGURE 20 – Opérateur **Reverse(2,5)**

Le pseudo-code de la méthode développée est présenté dans l'algorithme 2. On note DP l'algorithme de programmation dynamique permettant de résoudre le problème de fixation des dates à partir d'un ordre d'acquisitions fixé : celui-ci prend en argument un ordre d'acquisitions et renvoie la somme des récompenses individuelles correspondant aux instants d'observation optimaux, tout en mettant à jour les horaires effectifs de prise de vue pour chacune des acquisitions. On note également CESP (pour *Compute Earliest Start Plan*) la procédure qui renvoie, à partir d'un ordre d'acquisitions donné, un booléen *true* si le plan au plus tôt est réalisable, et *false* dans le cas contraire. Tous les tirages aléatoires se font suivant la loi uniforme.

Algorithm 2: Recherche locale pour la résolution du problème d'ordonnancement

Entrées : Une instance du problème d'ordonnancement (cf section 4.1)
Un nombre $nIter$ d'itérations pour l'algorithme
Initialisation : Obtention d'une solution initiale *currentSolution* via une des trois méthodes décrites en paragraphe 4.3.1
$currentReward \leftarrow DP(currentSolution)$
$bestSolution \leftarrow currentSolution$ $bestReward \leftarrow currentReward$
for $i{=}0\,;i{<}nIter\,;i{+}{+}$ **do**
  id choisi aléatoirement dans $[\![1;3]\!]$;
  i choisi aléatoirement parmi les indices des acquisitions
  $step$ choisi aléatoirement dans $[\![-3;3]\!]$
  if $id == 1$ **then**
    $testSolution \leftarrow Shift(currentSolution, i, i + step)$
  else
    if $id == 2$ **then**
      $testSolution \leftarrow Swap(currentSolution, i, i + step)$
    else
      $testSolution \leftarrow Reverse(currentSolution, i, i + step)$
    end
  end
  if $CESP(testSolution)$ **then**
    $currentSolution \leftarrow testSolution$
    $currentReward \leftarrow DP(currentSolution)$ **if**
     $currentReward > bestReward$ **then**
      $bestReward \leftarrow currentReward$
      $bestSolution \leftarrow currentSolution$
    end
  end
end
return bestSolution

4.4 Résultats numériques

Tout comme pour le problème de fixation des dates, les détails des tests réalisés sur le serveur de calcul de l'ONERA sont consignés ci-après.

4.4.1 Détails des instances

Comme pour les instances du problème de fixation des dates, l'orbite du satellite se situe à 500 km d'altitude. On a les paramètres $(\text{lat}_{\max}, \text{long}_{\max}) \in \{(32,1),(32,10),(4,10),(4,2)\}$, permettant d'aboutir aux archétypes d'instances décrits à la section 4.2. On fixe également $nIter = 500$, et on rappelle qu'on a imposé $n = 15$.

Un des problèmes liés à la résolution du problème d'ordonnancement est l'éventuelle rigidité des fenêtres temporelles : pour mettre en oeuvre l'algorithme 2, cela présuppose que les mouvements au sein de l'espace des solutions à partir des opérateurs définis ne sont pas trop restreints. Or expérimentalement, ces mouvements sont assez restreints pour un angle d'observation maximal défini de la même manière que pour le problème de fixation des dates. De ce fait, on effectue alors les tests également sur des instances où, pour chacune des acquisitions, l'angle maximal toléré est égal à 70^o. Cela a également nécessité une modification de la fonction de récompense, afin que la comparaison des valeurs des objectifs soit légitime : on définit désormais (avec $\alpha_i(t)$ et α_{prox} exprimés en $^\circ$) :

$$r_i^u(t) = \frac{0.9 * (70 - \alpha_i(t))}{70 - \alpha_{\text{prox}}}$$

et encore

$$r_i^e(t) = 1 - (1 - r_i^u(t))^2$$

De la sorte, cela n'entraîne pas de modification de la récompense pour des mêmes valeurs d'instants, ce qui est le cas si on laisse la dépendance en α_{prox}.

4.4.2 Résultats numériques

Les résultats numériques (temps de calcul et récompense totale) des tests décrits précédemment sont consignés ci-dessous. La taille de la liste d'acquisition initiale est constamment égale à 15. La ligne supérieure correspond aux instances où l'angle maximal toléré est, pour chaque acquisition, sélectionné aléatoirement parmi $\{40, 60\}^o$ (noté instances de type (A)), et la ligne inférieure aux instances où toute acquisition a un angle maximal toléré égal à 70^o (instances de type (B)).

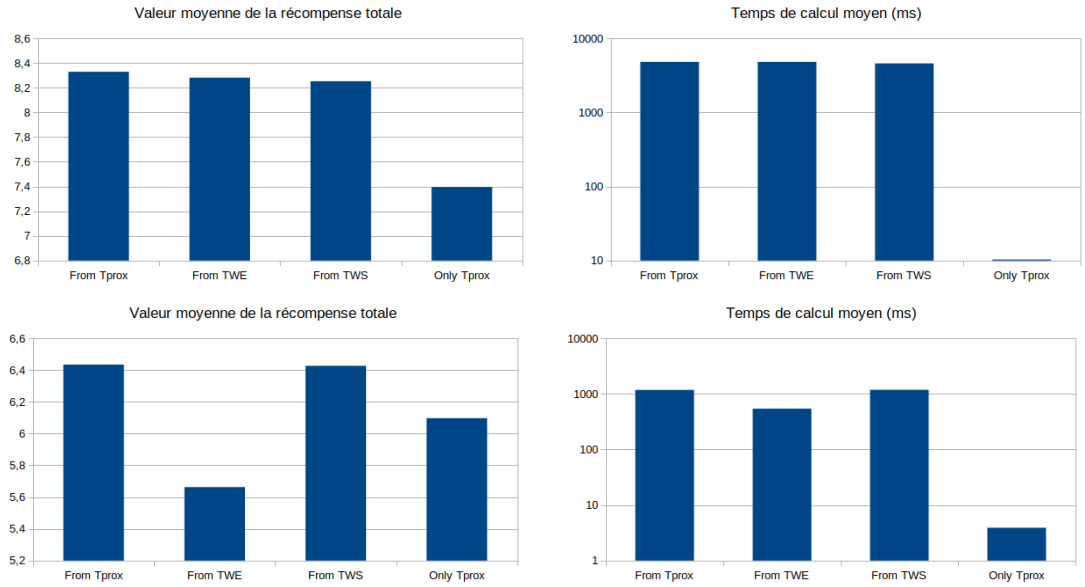

FIGURE 21 – Synthèse des résultats pour les instances de type (1)

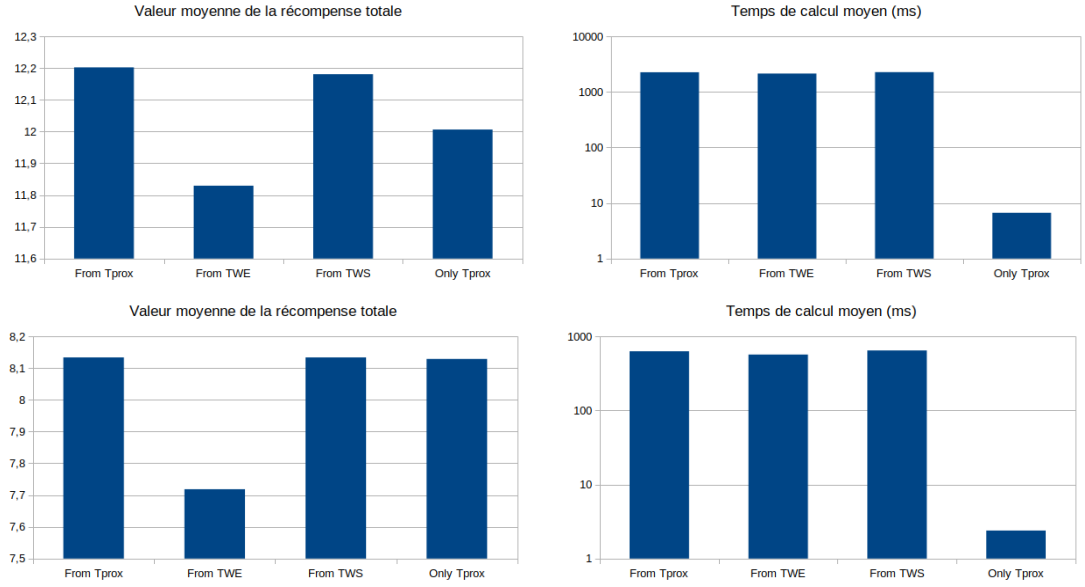

FIGURE 22 – Synthèse des résultats pour les instances de type (2)

Plusieurs éléments sont à remarquer :

— Sans surprise, les pires résultats sont observés sur les instances de type (3), quel que soit l'angle maximal toléré. En effet, la taille maximale de la liste d'acquisitions est notoirement plus petite en raison du trajet du satellite, du pôle Sud au pôle

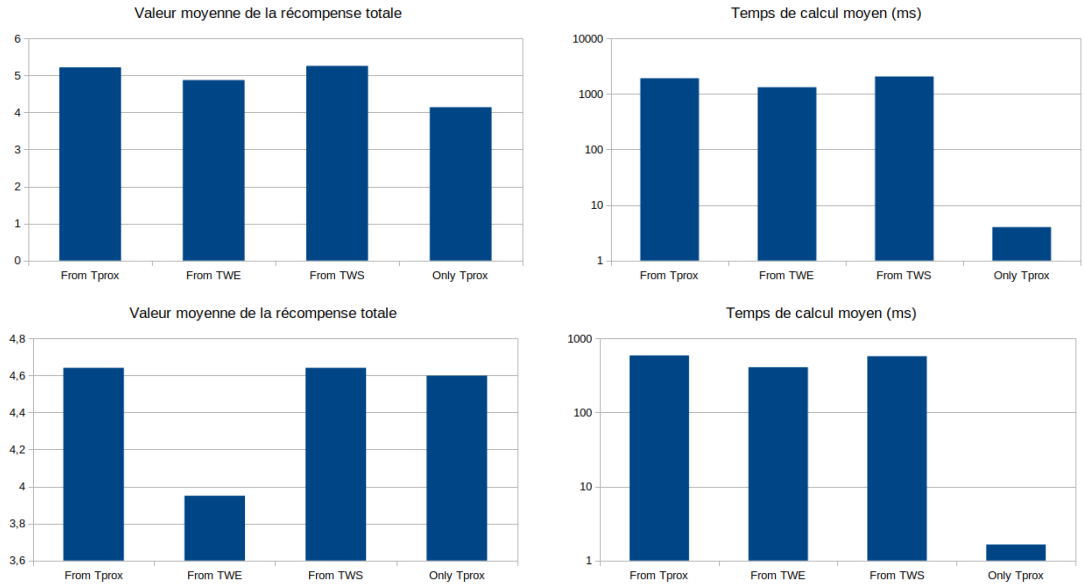

FIGURE 23 – Synthèse des résultats pour les instances de type (3)

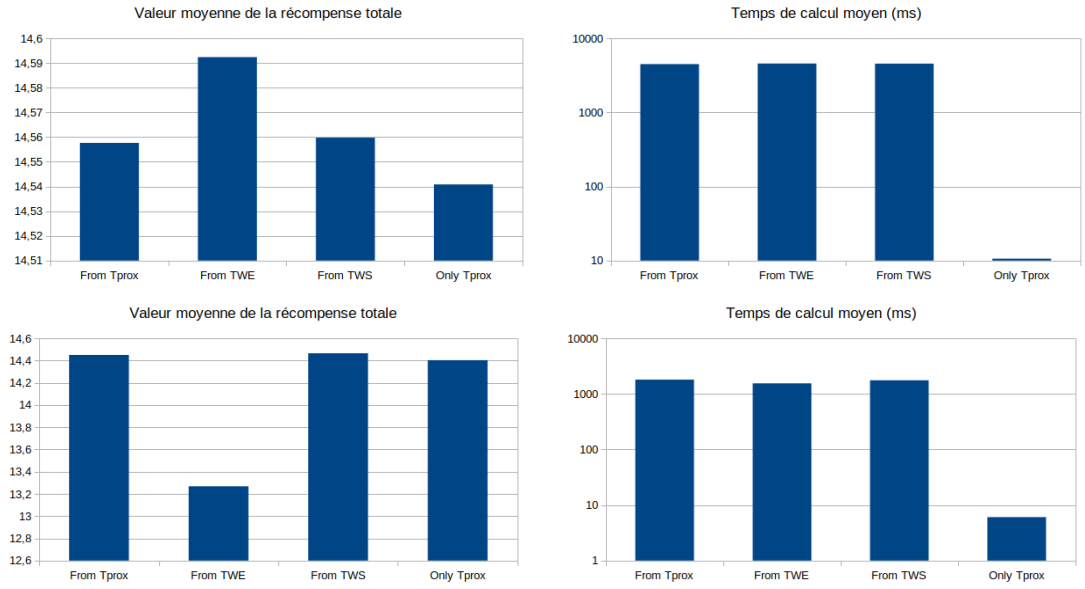

FIGURE 24 – Synthèse des résultats pour les instances de type (4)

Nord.

— Relativement aux autres, l'heuristique only tProx performe le mieux sur les instances de type (2) et (4), parfois même mieux que l'heuristique aléatoire en fonction de la solution initiale choisie. En particulier, pour les instances de type (4),

il n'est pas absurde de concevoir que le fait de ne pas respecter l'ordre croissant des temps de transition accrus, au risque de dégrader les angles d'observation des acquisitions. Or le temps de calcul est divisé par 1000 dans le cas de l'heuristique only tProx ; cela peut susciter de l'intérêt à l'appliquer dans le cas d'instances bien identifiées et correspondant à ces cas. On remarque en revanche un gap de 5à 10% sur les instances de type (1) : elle peine à être efficace pour des acquisitions très concentrées spatialement.

— En moyenne, le temps de calcul en passant des instances de type (A) à celles de type (B) est multiplié par 5, mais la valeur moyenne de l'objectif est notoirement améliorée (cette amélioration est plus discrète pour les instances de type (4)). Cela est effectivement du au fait que la construction des acquisitions dans les instances de type (A) permet de traiter davantage d'acquisitions.

— Sur la quasi-totalité des instances traitées, l'heuristique From Tprox est la plus performante des trois ; le tri initial des acquisitions par ordre croissant de t_{prox} semble donc faire sens comparé aux deux autres tris proposés.

Cela conclut l'analyse des résultats obtenus pour la résolution du problème d'ordonnancement.

5 Conclusion

Le projet LICHIE porte sur la conception et la gestion de constellations de satellites d'observation, il est donc naturel de s'intéresser à l'optimisation de la décision quand à l'attribution des prises de vue à ces satellites et leur ordonnancement pour chacun des satellites. Il s'agit d'un problème hautement combinatoire qu'il paraissait pertinent de décomposer en plusieurs couches de complexité. Pour le problème de fixation des dates, la méthode par programmation dynamique a fait ses preuves : elle aboutit dans la majorité des cas à la meilleure solution, le tout en un temps relativement restreint.

Le problème d'ordonnancement a fait l'objet d'une étude de manière conjointe à un approfondissement de la variété des instances, et si la recherche locale à voisinages multiples a pu démontrer l'incapacité de l'heuristique only tProx à traiter certains types d'instances, elle n'en reste pas moins relativement simpliste. Un premier pas vers une méthode plus évoluée repose sur une heuristique de meilleure insertion, utilisant une programmation dynamique bidirectionnelle inspirée par [4]. Cette heuristique prendra une partie du temps de la fin du stage.

Le problème de sélection au préalable des acquisitions a été mis de côté pour se concentrer sur l'ordonnancement et la fixation des dates de visite, de même que le traitement du problème pour une constellation entière d'EOS. On peut également remarquer que la couverture nuageuse, pouvant entraîner une grande dégradation de la qualité des prises de vue, n'a pas été abordée au cours de ce stage. Tous ces aspects, notamment le dernier, feront l'objet d'une thèse ONERA/CNES qui commencera en novembre prochain, incluant désormais des techniques d'apprentissage pour mieux anticiper l'impact des incertitudes météorologiques sur la qualité des décisions prises, à l'aide de données historiques.

Références

[1] Nabila Azi, Michel Gendreau, and Jean-Yves Potvin. An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers Operations Research*, 41 :167–173, 2014.

[2] Sara Maqrot, Stéphanie Roussel, Gauthier Picard, and Cédric Pralet. Orbit slot allocation in earth observation constellations. In *Conference on Prestigious Applications of Intelligent Systems (PAIS)*, volume 351 of *11th Conference on Prestigious Applications of Artificial Intelligence, 25 July 2022, Vienna, Austria (co-located with IJCAI-ECAI 2022)*, pages 3–16, Vienna, Austria, July 2022. IOS Press.

[3] Paul H. Morris, Robert A. Morris, Lina Khatib, Sailesh Ramakrishnan, and Andrew Bachmann. Strategies for global optimization of temporal preferences. In Mark Wallace, editor, *Principles and Practice of Constraint Programming - CP 2004, 10th International Conference, CP 2004, Toronto, Canada, September 27 - October 1, 2004, Proceedings*, volume 3258 of *Lecture Notes in Computer Science*, pages 408–422. Springer, 2004.

[4] Guansheng Peng, Reginald Dewil, Cédric Verbeeck, Aldy Gunawan, Lining Xing, and Pieter Vansteenwegen. Agile earth observation satellite scheduling : An orienteering problem with time-dependent profits and travel times. *Computers Operations Research*, 111 :84–98, 2019.

[5] Guansheng Peng, Guopeng Song, Lining Xing, Aldy Gunawan, and Pieter Vansteenwegen. An exact algorithm for agile earth observation satellite scheduling with time-dependent profits. *Computers Operations Research*, 120 :104946, 2020.

[6] Gauthier Picard. Auction-based and distributed optimization approaches for scheduling observations in satellite constellations with exclusive orbit portions. *CoRR*, abs/2106.03548, 2021.

[7] Samuel Squillaci, Stéphanie Roussel, and Cédric Pralet. Parallel Scheduling of Complex Requests for a Constellation of Earth Observing Satellites. In *PAIS 2022*, Frontiers in Artificial Intelligence and Applications, Vienne (AUT), France, July 2022. IOS Press.

[8] Vincent F. Yu, Parida Jewpanya, Shih-Wei Lin, and A.A.N. Perwira Redi. Team orienteering problem with time windows and time-dependent scores. *Computers Industrial Engineering*, 127 :213–224, 2019.