

Optimisation des acquisitions et des vidages de données pour des satellites d'observation de la Terre

Projet ISAE COS

C. Pralet

2017-2018

ONERA

THE FRENCH AEROSPACE LAB

retour sur innovation

Programmation mission pour:

- ▶ un système de type Pléiades: 2 satellites agiles sur la même orbite, à 180° l'un de l'autre (orbite héliosynchrone, 694 km d'altitude)
- ▶ en utilisant un réseau de 6 stations sol du CNES

Objectif: produire au sol

- ▶ un **plan d'acquisition** pour chaque satellite
- ▶ un **plan de vidage de données** (transmission des acquisitions réalisées vers les stations sol disponibles)

Plans à produire sur une **durée de 24h** (un plan d'une durée de 24h envoyé toutes les 8h au satellite, i.e. planification sur un horizon glissant)

Données statiques (constantes au cours de la durée de vie du satellite):

- ▶ un ensemble d'**utilisateurs** ayant contribué au satellite, avec pour chaque utilisateur un quota d'utilisation $\in [0, 1]$ fonction de sa contribution
- ▶ un ensemble de **satellites** (deux satellites ici), avec pour chaque satellite la liste de ses paramètres orbitaux
- ▶ un ensemble de **stations sol** (6 stations du réseau CNES), avec pour chacune ses coordonnées géographiques

Données toutes regroupées dans un **fichier XML `scenario.xml`**

Parser déjà disponible pour lire ces données et générer des objets Java correspondants

Données dynamiques (qui changeront au cours de la durée de vie du satellite):

- ▶ l'**horizon de planification** (de durée potentiellement plus faible que 24h pour réaliser des tests)
- ▶ le **catalogue des acquisitions candidates** sur l'horizon de planification (demandes d'acquisition formulées par les utilisateurs)
- ▶ le **catalogue des acquisitions déjà enregistrées** à bord du satellite au début de l'horizon de planification (acquisitions déjà réalisées mais pas encore vidées vers des stations sol)

Caractéristiques des acquisitions candidates:

- ▶ un **identifiant** unique (entier) et un nom
- ▶ un **niveau de priorité**: 0 (le plus important) ou 1 (le moins important)
- ▶ l'**utilisateur** demandant l'acquisition
- ▶ les coordonnées géographiques de la zone concernée
- ▶ la liste des **opportunités d'acquisition**, correspondant aux fenêtres temporelles de visibilité de la zone, avec pour chaque fenêtre:
 - ▶ le **satellite** associé à la visibilité
 - ▶ les **dates de début et de fin** de la fenêtre de visibilité
 - ▶ la **durée d'acquisition** si l'observation est réalisée dans cette fenêtre
 - ▶ l'**angle zenith-satellite** obtenu dans la fenêtre lorsque le satellite est au plus près de la zone à imager (qualité image fonction de cet angle)
 - ▶ l'**angle de roulis** requis par le satellite pour réaliser l'acquisition
 - ▶ la **probabilité de présence de nuages** sur la zone à observer (image potentiellement inexploitable si présence de nuages)
 - ▶ le **volume mémoire** occupé si l'acquisition est réalisée

Caractéristiques des acquisitions déjà enregistrées:

- ▶ un **identifiant unique** (entier)
- ▶ le **satellite** sur lequel l'acquisition est enregistrée
- ▶ l'**utilisateur** associé à l'acquisition
- ▶ le **niveau de priorité** de l'acquisition
- ▶ la **date de réalisation** de l'acquisition
- ▶ le **volume mémoire occupé** par l'acquisition

Egalement une liste de fenêtres temporelles disponibles pour réaliser des vidages de données (appelées des **fenêtres de visibilité station**), avec pour chacune:

- ▶ un **identifiant unique** (entier)
- ▶ le **satellite** et la **station** associés à la fenêtre de visibilité station
- ▶ les **dates de début et de fin** de la fenêtre

Données d'entrée “dynamiques” toutes regroupées dans un **fichier XML**

Parser déjà implémenté pour lire le fichier XML et générer des objets Java représentant ces données

Décisions à prendre pour construire un plan d'acquisition:

- ▶ sélection des acquisitions à réaliser parmi les acquisitions candidates
- ▶ pour chaque acquisition retenue, sélection d'un satellite chargé de la réaliser
- ▶ pour chaque acquisition retenue, sélection d'une date de réalisation

Décisions à prendre pour construire un plan de vidage:

- ▶ pour chaque satellite, sélection des vidages à réaliser parmi les vidages candidats
- ▶ pour chaque vidage retenu, sélection d'une station sur laquelle le réaliser
- ▶ pour chaque vidage retenu, sélection d'une date de début de vidage

Quelques **contraintes**:

- ▶ acquisition uniquement pendant les **fenêtres de visibilité des zones**
- ▶ vidage uniquement pendant les **fenêtres de visibilité station**
- ▶ pour un satellite donné, **au plus une acquisition réalisée à chaque instant** (un seul instrument d'acquisition)
- ▶ **durée de transition** requise entre deux acquisitions successives (besoin de changer l'attitude du satellite pour passer de la 1ère à la 2nde)
Hypothèse simplificatrice: durée de transition fonction de l'écart en termes d'angle de roulis et d'une vitesse angulaire moyenne en roulis du satellite
- ▶ pour un satellite donné, **au plus un vidage de données réalisé à chaque instant** (un seul canal d'émission de données vers le sol)
- ▶ pour une station donnée, **au plus une acquisition transmise** à chaque instant (hypothèse d'une seule antenne sol au niveau de chaque station)
- ▶ date de début de vidage des données d'une acquisition située **après** la date de réalisation de l'acquisition associée

“Quelques” **critères** utilisables:

- ▶ **priorité** des acquisitions réalisées (une acquisition de priorité 0 toujours préférée à n'importe quel groupe d'acquisitions de priorité 1)
- ▶ **partage** de la ressource satellite entre les différents utilisateurs
- ▶ **qualité des images** réalisées (préférence pour les prises de vue le plus possible à la verticale des zones à imager)
- ▶ préférence pour les plans d'acquisition **maximisant le nombre d'images exploitables** (prise en compte de la couverture nuageuse)
- ▶ préférence pour les plans de vidages qui réservent **le moins de créneaux de vidage** sur des stations sol
- ▶ préférence pour la réalisation des **acquisitions au plus tôt**
- ▶ préférence pour la réalisation des **vidages au plus tôt**
- ▶ préférence pour les plans d'acquisition **sollicitant le moins les actionneurs gyroscopiques**

Dans les **fichiers Java fournis**, déjà des exemples montrant comme réaliser une chaîne d'optimisation pour obtenir des plans d'acquisition et de vidage

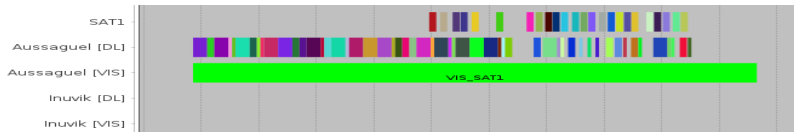
Fichier **BadAcquisitionPlanner.java**:

- ▶ lecture du scénario
- ▶ génération de fichiers OPL: un fichier .mod contenant un modèle du problème d'optimisation des acquisitions, et un fichier .dat par satellite
- ▶ résolution à l'aide de CPLEX
- ▶ plan d'acquisition trouvé écrit dans un fichier (format prédéfini)

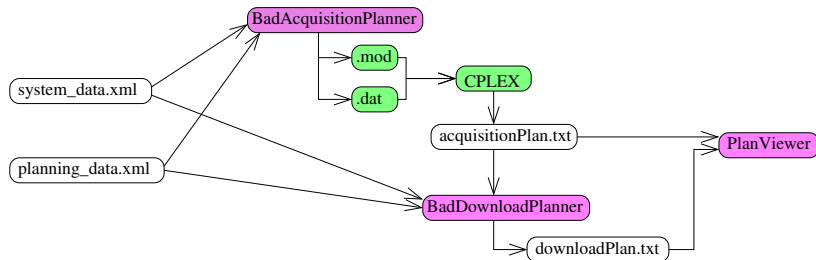
Fichier **BadDownloadPlanner.java**:

- ▶ lecture du scénario et du plan d'acquisition généré à l'étape précédente
- ▶ application d'une règle de décision qui insère des vidages dans un ordre chronologique (parcours des acquisitions à vider par date de réalisation croissante et parcours des visibilité station dans l'ordre chronologique)
- ▶ plan de vidage trouvé écrit dans un fichier texte (format prédéfini)

Outil de visualisation des plans prenant en entrée les fichiers de scénario, les fichiers de plan d'acquisition, les fichiers de plan de vidage



Chaîne d'optimisation obtenue avec les “BadPlanners”:



Approche générale:

- ▶ **sujet ouvert**, avec une grande liberté sur les modèles et méthodes utilisables (pas de corrigé type, pas de question 1, question 2...)
- ▶ pratique de toute la **chaîne de l'optimisation**, allant de la formalisation des données d'entrée à la visualisation et l'analyse des solutions produites
- ▶ besoin de s'**interfacer** avec un code déjà existant

Logistique:

- ▶ **11h de projet réparties sur 3 séances** (3h + 4h + 4h)
- ▶ un “**sprint review**” (bilan d'avancement) au milieu de chaque séance, pour discuter des idées, s'inspirer de bonnes pratiques d'autres groupes...

Note: on ne s'attend pas à ce qu'en 11h tous les aspects du problème soient traités! → à vous de choisir comment vous faites évoluer votre sujet

69 étudiants, deux options de découpage en groupes:

- ▶ soit **15 groupes de 4 + 3 groupes de 3**,
- ▶ soit **16 groupes de 4 + 1 groupe de 5**

Fonctionnement conseillé pour un groupe:

- ▶ se **partager clairement le travail** au sein d'un groupe (élèves d'un même groupe pas tous sur le même sous-problème avec la même technique, faire par exemple 2 sous-groupes de 2 pour un groupe de 4)
- ▶ élèves d'un même groupe tous dans la même partie de la salle informatique

Fonctionnement entre groupes:

- ▶ possibilité de récupérer un morceau de code d'un autre groupe pour résoudre un sous-problème particulier, **mais dans ce cas obligation de faire une référence explicite au groupe fournisseur du code**

Phase finale:

- ▶ **soutenance de projet le 10 novembre**: 15-20 minutes par groupe, objectif de défendre les qualités de votre système de programmation mission
- ▶ **rapport de projet au format pdf + une archive contenant le code source**, à remettre sur un dépôt sur le LMS pour le **24 novembre**

Contenu attendu du rapport (**20 pages max**):

- ▶ description de l'**utilisation de techniques** d'optimisation (modèles développés et méthodes d'optimisation utilisées)
- ▶ description de l'**organisation du code** source
- ▶ analyse des **performances** en termes de critères et des temps de calcul

Etape 1: faire tourner la chaîne d'optimisation déjà implémentée

Etape 2: définir des critères d'optimisation et intégrer leur calcul au code fourni

Etape 3: construction de votre système de programmation mission

Quelques idées en vrac (liste non exhaustive):

- ▶ planification pour chaque satellite ou planification pour la constellation
- ▶ optimisation de l'enchaînement d'un ensemble d'acquisitions fixé
- ▶ optimisation du centrage des acquisitions pour un ordre d'acquisition fixé
- ▶ dates de réalisation des activités recherchées dans des intervalles continus ou discrétisation en considérant seulement un nombre restreint de dates
- ▶ traitement simultané acquisition + vidage ou traitement en séquence
- ▶ optimisation orbite par orbite
- ▶ test de plusieurs approches de résolution (programmation par contraintes, programmation linéaire en nombre entiers, règles de décision...)
- ▶ réflexion sur l'optimisation multi-critère (somme pondérée, hiérarchie...)