Dimensionnement d'une chaîne d'assemblage de satellites avec plusieurs cadences sous contraintes de régularité

Anouck Chan[1], Stéphanie Roussel[1], Thomas Polacsek[1]

ONERA/DTIS, Université de Toulouse

prenom.nom@onera.fr

Mots-clés : *Chaîne d'assemblage, constellations de satellites, ordonnancement, multi-cadences, régularité*

1 Introduction

Avec l'émergence des constellations de satellites de ces dernières années, les chaînes d'assemblage de satellites doivent répondre à de nouvelles demandes : fabriquer plusieurs exemplaires d'un même satellite sur des cadences très courtes (de l'ordre du mois), gérer plusieurs cadences de fabrication pour différentes constellations, mais aussi continuer à fabriquer des satellites spécifiques en un seul exemplaire sur des horizons plus longs (de l'ordre de l'année). L'assemblage de ces satellites se fait au sein d'une infrastructure existante et avec des machines et outils spécifiques nécessaires (dalles asismisque, environnement de type salle blanche, *etc.*) qui doivent être partagés entre les activités d'assemblage de tous les satellites. Dans ce travail, nous nous intéressons au dimensionnement d'une chaîne d'assemblage et cherchons plus précisément à décider si une chaîne avec une infrastructure et un nombre de machines donnés est en capacité de répondre à un carnet de livraison sur un horizon temporel. Nous proposons une approche basée sur la programmation par contraintes pour calculer un ordonnancement des activités d'assemblage dans lequel nous minimisons les machines utilisées et pour lequel des contraintes de régularité sont satisfaites. Nous évaluons cette approche sur des scénarios réels fournis par un industriel.

2 Formalisation du problème

Le problème peut être vu comme une variante du RCPSP (*Resource Constrained Project Scheduling Problem*). Plus précisément, nous disposons d'un ensemble de machines avec pour chacune une unique compétence associée. Chaque compétence possède une capacité qui représente le nombre d'activités pouvant être simultanément réalisées par chaque machine. Plusieurs types de satellites sont produits sur la chaîne. Pour un type de satellite donné, on dispose du nombre d'exemplaires devant être assemblés et de la cadence d'assemblage. Cette dernière permet de calculer une date de livraison pour chaque exemplaire. À chaque type de satellite, est également associé un ensemble d'activités d'assemblage. Chaque activité est caractérisée par une durée et des compétences nécessaires pour la réaliser. Les activités d'un même type de satellite sont soumises à des contraintes de précédence. Ces contraintes peuvent être représentées par un graphe dirigé acyclique.

En plus des contraintes classiques inhérentes au respect des précédences et des caractéristiques des ressources, nous introduisons deux nouvelles contraintes. La première impose le nombre maximum d'exemplaires pouvant être produits simultanément. La seconde consiste en des contraintes de régularité sur l'ordonnancement produit. Intuitivement, la régularité signifie que les ordonnancements des activités pour les exemplaires d'un même type de satellite sont similaires. Nous avons défini plusieurs types de régularité. Ainsi, la régularité stricte consiste

à avoir des ordonnancements identiques pour chaque exemplaire d'un satellite, modulo un décalage fixe des dates de début de chaque activité. On se ramène alors à des variantes de RCPSP cycliques ([1]). Des régularités plus souples consistent à imposer que l'ordre de début des activités soient les mêmes d'un exemplaire à l'autre (régularité dite *ordinale*). Finalement, on cherche à dimensionner la chaîne du point de vue des machines utilisées. Pour cela, on associe à certaines machines un coût non nul, indiquant que l'on préfère ne pas les utiliser, et on minimise le coût total.

Nous avons traduit le problème en programmation par contraintes. Pour chaque activité de chaque exemplaire de chaque type de satellite, nous considérons une variable intervalle (langage OPL de CpOptimizer). Comme plusieurs machines ont la même compétence, il existe plusieurs alternatives (ou *modes*) pour réaliser chaque activité. Chaque mode est encodé par une variable intervalle optionnelle, *i.e.* qui n'est pas obligatoirement présent dans l'ordonnancement final.

3 Expérimentations

Nous disposons de jeux de données fournis par un industriel. Nous avons trois types de satellites, 90 machines et considérons les activités liées à l'assemblage de l'instrument optique équipant ces satellites (entre 10 et 50 activités selon le type de satellite). Plusieurs scénarios de livraison sont considérés (entre 2 et 5 exemplaires pour chaque type de satellite) sur un horizon temporel de plusieurs mois. Nous avons utilisé le solveur CpOptimizer 20.1 pour l'évaluation.

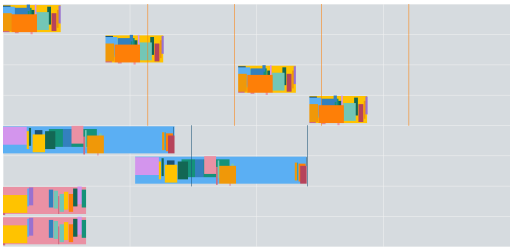

FIG. 1 – Régularité stricte

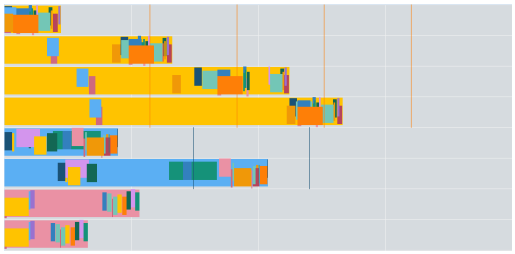

FIG. 2 – Régularité ordinale

Les figures 1 et 2 illustrent les ordonnancements obtenus respectivement avec la contrainte de régularité stricte et la contrainte de régularité ordinale. Les expérimentations réalisées montrent que l'approche peine à trouver des solutions lorsque l'on augmente le nombre d'activités de chaque satellite et que l'on considère la régularité stricte (plusieurs jours de calcul).

4 Perspectives

Une première perspective consiste à définir d'autres types de régularité, correspondant précisément aux besoins des industriels, *e.g.* régularité ordinale sur des sous-ensembles de tâches. Une deuxième perspective consiste à travailler sur d'autres encodages en programmation par contraintes et d'utiliser différents solveurs, par exemple au travers de PyCSP3 [2].

Remerciements Ce travail a été réalisé avec le soutien de la BPI dans le cadre du projet PSPC LiChIE (Lion Chaîne Image Élargie), coordonné par Airbus Defense and Space.

Références

[1] Alessio Bonfietti, Michele Lombardi, Luca Benini, and Michela Milano. A Constraint Based Approach to Cyclic RCPSP. In J. Lee, editor, *Principles and Practice of Constraint Programming – CP 2011*, pages 130–144, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[2] Christophe Lecoutre and Nicolas Szczepanski. Pycsp3 : Modeling combinatorial constrained problems in python, 2020.