Le problème de vendange sélective : une approche Programmation Par Contraintes

G. Volte[1], É. Bourreau[1], R. Giroudeau[1], O. Naud[2]

[1] LIRMM, Université de Montpellier, CNRS, Montpellier, France
{volte,giroudeau,bourreau}@lirmm.fr

[2] ITAP, Irstea, Montpellier SupAgro, Univ Montpellier, Montpellier, France
olivier.naud@irstea.com

1 Définition du problème

Le problème de la vendange sélective (Differential Harvest Problem en anglais, [2]) consiste à optimiser le temps de récolte tout en récoltant dans une benne spécifique une quantité minimum donnée de raisins de la meilleure qualité. Il peut être vu comme un problème de cVRP enrichi avec des contraintes métiers. On considère deux qualités de raisin, que nous désignerons par A pour la meilleure qualité et B pour la qualité inférieure. *Rmin* désigne la quantité minimum de raisin A à récolter de façon sélective. Grâce à une information agronomique obtenue à priori, il est possible de cartographier (cf. Figure 1) une parcelle de vigne, de n rangs, prête à être vendanger en distinguant des zones en fonction de la qualité des raisins. Il existe des vendangeuses géolocalisées munies de deux trémies (réservoirs pour la récolte), chacune de capacité *Cmax*, capables d'utiliser une telle carte pour séparer les raisins de type A du raisin de type B. Lorsque l'une des deux trémies est pleine il faut aller vider les deux dans une benne située au bord de la parcelle.

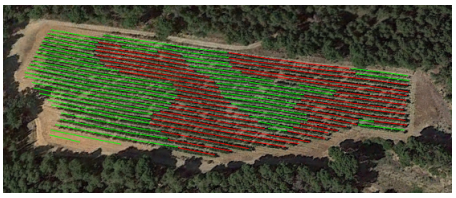

FIG. 1 – Figure illustrant les différentes zones de qualité de vigne.

Ce problème a été introduit par [2], qui a essayé de le résoudre le problème avec la programmation par contraintes. Cependant l' approche qui a été utilisée se basait sur la création d'une contrainte globale qui modélise le problème.

2 Programmation par contraintes

Notre approche de programmation par contraintes est basé sur les modèles à vocation générique décris dans le livre [1]. Nous considérons une modélisation classique prédécesseurs/successeurs mais en employant des contraintes globales réputées efficaces, plus fines, pour représenter les contraintes de capacité des véhicules.

Les contraintes de capacité des véhicules sont modélisées avec la contrainte globale *cumulative* : son rôle est d'ordonnancer des tâches sur un horizon (pas forcément temporel) sans superposition en vérifiant qu'aucune tâche ne dépasse une certaine hauteur. Pour notre problème, cette contrainte est une variante du problème de Bin packing où l'on doit placer des objets (les rangs de la vigne) de hauteur variables (la quantité récoltée) dans des bin (le véhicule) de tailles *Cmax*, chaque bin représente un véhicule et l'affectation d'un objet à un véhicule représente le fait que le rang correspondant à l'objet appartienne à la tournée du véhicule. Pour cela, nous définissons un ensemble de tâches $Task^{qA}$ pour les raisins de qualité A et $Task^{qB}$ pour ceux de qualité B. L'affectation de ces tâches aux véhicules se fait en fonction

de la qualité de raisins récoltés et de la sélectivité du véhicule (voir Figure 2). Les positions paires représentent les tâches récoltant du raisin de qualité A et les positions impaires le raisin de qualité B. Pour chaque véhicule, il y a quatre positions possibles les deux premières pour la récolte en mode sélectif et les deux autres en non-sélectif.

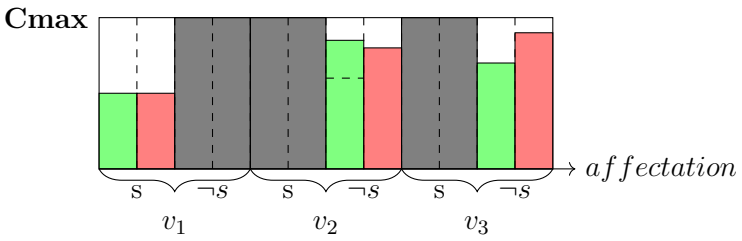

FIG. 2 – Représentation des contraintes de capacités des véhicules avec la contrainte *cumulative*.

La position d'une tâche nous donne la qualité du raisin récoltée et sa hauteur la quantité de raisin récoltée. La quantité de raisin récoltée dépend du sens de parcours de la rangée et de la sélectivité du véhicule.

2.1 Stratégie utilisée : le Snake

Le Snake est une stratégie qui construit totalement une route puis passe à la suivante et ainsi de suite. L'intérêt est de trouver une bonne solution en peu de nœuds de branchement. En général, les autres stratégies construisent des fragments de routes et en bas de l'arbre de décision essaient d'assembler ces fragments (difficile sachant que de nombreuses possibilités d'arrangement sont possibles) pour obtenir des routes valides.

Le fonctionnement de cette stratégie est le suivant (voir Figure 3) :

La première variable est le premier dépôt initial disponible. Le choix de la nouvelle variable est déterminé par le choix de la valeur de la variable précédemment instantiée. La première valeur disponible pour la variable est sélectionnée, les valeurs sont triées de manière croissante (on souhaite minimiser les distances). Une fois qu'un dépôt final est instantié, la prochaine variable sélectionnée est le dépôt initial suivant. Les variables utilisées sont des variables calculant la distance vers le successeur. De plus, ces variables sont triées afin que l'optimisation commence par des routes sélectives.

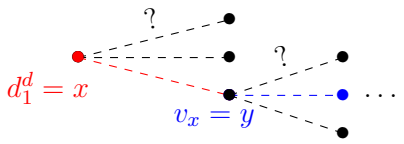

FIG. 3 – Illustration de la stratégie Snake.

En ajoutant cette stratégie constructive nous espérons obtenir de bon résultats dans l'optique d'hybrider la programmation par contraintes avec des techniques de génération de colonnes. Des tests (en cours sur des instances réelles et simulées) seront présentés.

Références

[1] E. Bourreau, M. Gondran, P. Lacomme, and M. Vinot. *De la Programmation Linéaire à la Programmation Par Contraintes*. 2019.

[2] N. Briot, C. Bessiere, and P. Vismara. A constraint-based approach to the differential harvest problem. In Gilles Pesant, editor, *Principles and Practice of Constraint Programming*, pages 541–556, Cham, 2015. Springer International Publishing.