Séquencement des Travaux de Collecte de Prix avec Une Ressource Commune et Plusieurs Ressources Secondaires : une approche fondée sur les variables de séquence

Augustin Delecluse[1,2*] **Pierre Schaus**[1]

[1] UCLouvain, Belgique
[2] TRAIL, Belgique
{augustin.delecluse,pierre.schaus}@uclouvain.be

Résumé

Le séquencement des travaux de collecte de prix avec une ressource commune et plusieurs ressources secondaires est un problème de planification dont l'objectif consiste à maximiser le prix d'exécution de tâches, soumises à plusieurs contraintes. Les approches actuelles de résolution se basent sur des modèles en programme linéraires à nombres entiers mixtes, de la recherche locale ou encore de la programmation dynamique. Ce papier présente des résultats préliminaires sur une méthode de résolution en programmation par contraintes, s'appuyant sur des variables de séquence traditionnellement employée dans des problèmes d'itinéraires de véhicules. Couplée à de la recherche à large voisinage, elle permet une résolution efficace d'instances de grandes tailles.

Abstract

The Prize-Collecting Job Sequencing with One Common and Multiple Secondary Resources is a scheduling problem whose objective is to maximize the execution price of tasks subject to several constraints. Current approaches to solving this problem are based on mixed integer linear program models, local search or dynamic programming. This paper presents preliminary results on a constraint programming method based on sequence variables traditionally used in vehicle routing problems.

1 Introduction

Le Séquencement des Travaux de Collecte de Prix avec Une Ressource Commune et Plusieurs Ressources Secondaires (STCP-URCPRS) (Prize-Collecting Job Sequencing with One Common and Multiple Secondary Resources - PC-JSOCMSR - en anglais) est un problème de planification dont l'objectif consiste à maximiser le prix d'exécution de tâches, soumises à plusieurs contraintes. Il a été introduit dans [4], motivé par la planification de patients nécessitant un thérapie par particules comme traitement contre le cancer. Ce problème comporte une série de n tâches $t \in T$, chacune possédant une durée d_t pour son exécution ainsi qu'une récompense z_t si la tâche est exécutée. Chaque exécution requiert une ressource globale ainsi qu'une ressource secondaire associée $R_i \in R$, avec R le set de ressources associées disponibles dans le problème. La ressource globale et les ressources secondaires ne peuvent se consacrer qu'à une seule tâche à la fois : aucun chevauchement de tâches n'est possible. De plus, l'exécution d'une tâche t est davantage contrainte sur une ressource secondaire : elle comporte un durée de pré-traitement d_t^{pre} et post-traitement d_t^{post} durant laquelle la ressource secondaire n'est pas disponible. En outre, l'exécution d'une tâche t ne peut pas être interrompue à mi-chemin, et doit être faite dans l'une de ses fenêtres de temps spécifiées $w_t = \{w_{tk} \mid k = 0, \dots, \omega_t\}$, avec $w_{tk} = [w_{tk}^{start}, w_{tk}^{end}]$. Le but consiste à maximiser la somme des récompenses des tâches exécutées. Un exemple de solution faisable au problème est présenté dans la Figure 1.

Les résolutions les plus efficaces se font via des approches en programme linéraires à nombres entiers mixtes ou de la recherche locale pour des instances de grandes taille, lorsque la preuve d'optimalité n'est

*Papier doctorant : Augustin Delecluse[1,2] est auteur principal.

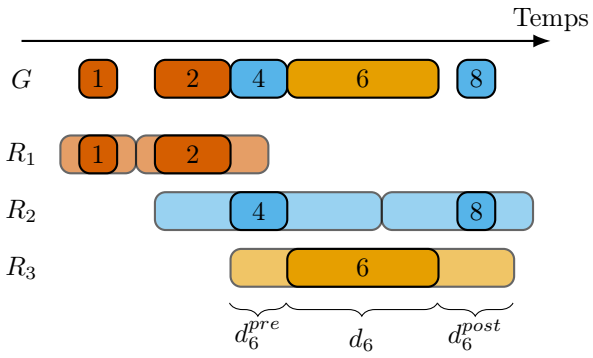

FIGURE 1 – Exemple de solution faisable pour le STCP-URCPRS. Les tâches exécutées sont les tâches 1, 2, 4, 6 et 8. En plus de nécessiter la ressource globale G pendant une durée d_t, chaque tâche t consomme également sa ressource secondaire associée q_t avec un temps de pré-traitement d_t^{pre} et de post-traitement d_t^{post}. La valeur objective est la somme des récompenses : $z_1 + z_2 + z_4 + z_6 + z_8$.

pas de mise [2]. D'autres méthodes, en programmation dynamique, diagrammes de décisions ou programmation par contraintes (PPC) ont également été développées [3]. Nous proposons une nouvelle approche se vasant sur des Variables de Séquence, habituellement employées pour des problèmes de routage de véhicules ([1, 5]). Celles-ci sont introduites dans la section 2. Un modèle les employant pour résoudre le STCP-URCPRS est présenté dans la section 3. Les performances sont évaluées dans la section 4.

2 Variables de séquence

La variable de séquence utilisée dans notre approche correspond à celle de [1], elle-même basée sur [5]. Certaines des notations introduites dans ces travaux sont repris dans cette section.

Étant donné un graphe dirigé $G(V, E)$ sur un ensemble de nœuds V et d'arrêtes E entre les nœuds, une variable de séquence, notée Sq, est une variable représentant l'ensemble des ordonnancements possibles sur des sous-ensembles des nœuds de V. Plus précisément, elle partitionne V en trois ensembles disjoints : les nœuds devant être visités par la séquence (R), les nœuds possiblement visités par la séquence (P) et les nœuds exclus de la séquence (X). La variable définit un ordonnancement sur des nœuds $S \subseteq R$ insérés dans la route et actuellement visités. La variable maintient pour chaque nœud $v \in V$ un ensemble d'arrêtes servant à représenter la séquence. Les arrêtes sont soit utilisées pour représenter l'ordonnancement actuel des nœuds, soit pour permettre d'étendre la séquence courante via des *insertions*.

Une insertion consiste en l'ajout d'un nœud $x \in V$ entre deux nœuds successifs u, v actuellement dans la séquence courante. Elle a pour effet de requérir le nœuds s'il ne l'était pas déjà ($x \in R \wedge x \notin P$) et de rajouter le nœud entre les deux nœuds successifs dans l'ordonnancement actuel. Plusieurs invariants permettent de détecter et retirer des arrêtes ne permettant pas d'étendre la séquence, par exemple pour le cas d'un nœud ne comportant que des arrêtes entrantes mais aucune sortante, ce qui ne permet pas de faire un détour par ce nœud [1]. Un exemple de variable de séquence se trouve dans la Figure 2.

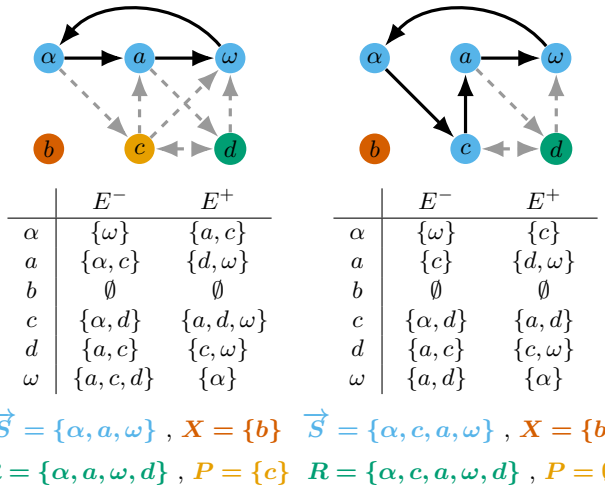

FIGURE 2 – Représentation d'une variable de séquence. La partie gauche montre une variable de séquence visitant les nœuds α, a et puis ω, et pouvant être étendue via des arrêtes dans le graphe (lignes en pointillés). En-dessous se trouve une tâble montrant les arrêtes entrantes E^- et sortantes E^+ des nœuds et le partage des nœuds entre R, P et X. De plus, une liste ordonnée $\overrightarrow{S}$ dénote l'ordonnancement actuel de la séquence. La partie droite montre une modification où le nœud c a été inséré entre les nœuds α et ω, changeant son statut et enlevant les arrêtes (c, ω) et (α, a) car elles sont détectées comme ne pouvant pas servir à étendre la séquence via des détours.

Le domaine $D(Sq)$ d'une variable de séquence capture toutes les séquences construites par insertions de nœuds, et visitant l'ensemble des nœuds requis ($v \in R$). Les modifications du domaine permettent d'insérer des nœuds, d'en requérir de nouveaux, d'en exclure et d'enlever des arrêtes, le tout en temps constant ou linéaire. Les contraintes appliquées sur ces variables utilisent ces mêmes opérations, que ce soit en enlevant des arrêtes qui, employées dans un détour, génè-

reraient une contradiction dans le problème, ou bien en excluant des nœuds détectés comme impossibles à visiter. Nous listons brièvement quelques contraintes d'intérêt pour ce papier, tout en référant le lecteur à [1] pour le fonctionnement des algorithmes de filtrage.

- `CollecteDePrix`(Sq, Z, z), liant une variable de séquence Sq à une variable entière Z.
- `SousSéquence`(Sq_1, Sq_2), imposant qu'une séquence Sq_2 doit être une sous-séquence d'une super-séquence Sq_1 : chaque nœud visité dans la sous-séquence est également visité dans la super-séquence, tout en préservant l'ordre de visite.
- `TempsDeTransition`(Sq, D, d, Tr) lie des variables entières D de début de fenêtre de temps, définies pour chaque nœud, à des visites effectuées par la séquence. Étant donné une durée de visite d pour un nœud et une matrice de temps de transition Tr entre nœuds, elle s'assure que les temps de visite des nœuds respecte les transitions temporelles. Elle est définie mathématiquement dans l'équation (1), qui décrit que le temps de début de la visite d'un nœud v doit se faire après la visite de son prédécesseur u et du temps de trajet nécessaire pour effectuer une tâche en u et effectuer la transition $(d_u + Tr_{u,v})$.

$$D_u + d_u + Tr_{u,v} \leq D_v \tag{1}$$

- `LienDeNœud`(Sq_1, Sq_2, v) : garantit que le statut d'un nœud v est le même dans deux variables de séquence Sq_1 et Sq_2. Dès que le nœud v devient exclus ou requis dans une séquence, il le devient également dans l'autre.

3 Méthode de résolution

Nous présentons d'abord le modèle en PPC suivi de la procédure de recherche.

3.1 Modèle

Le problème est modélisé avec une variable de séquence G représentant les tâches effectuées sur la ressource globale, et une variable de séquence S_i par ressource secondaire $Ri \in R$. La variable de séquence G comporte un nœud par tâche dans le problème, tandis que les variables associées aux ressources secondaires ont toutes les tâches qui ne sont pas reliées à leur machine exclues de leur domaine. Les fenêtres de temps de chaque tâche t sont représentées par des variables entières D_t indiquant le temps de début de la tâche t, dont les domaines sont restreint aux temps disponibles w_t pour l'exécution de la tâche correspondante. Une dernière variable Z capture la somme des prix d'exécution des tâches.

Certains tâbleaux et matrices sont calculés avant l'ajout de contraintes globales, afin de faciliter leur usage et d'obtenir une meilleure efficacité lors de la résolution. En particulier, la contrainte `TempsDeTransition` nécessite une matrice de temps de transition entre nœuds. Deux matrices sont donc initialisées : $O \in \mathbb{Z}^{T \times T}$ représente les temps de transitions en cas de changement de tâche sur la machine globale, chaque entrée de la matrice étant nulle. La matrice $Tr \in \mathbb{Z}^{T \times T}$, au contraire, correspond aux temps de transitions lors du changement de tâches sur une ressource secondaire. Ses entrées sont définies par $Tr_{ij} = d_i^{post} + d_j^{pre}$. Finalement, T_i dénote les tâches associées à la ressource secondaire R_i : $T_i = \{t \in T \mid q_j = R_i\}$.

Le modèle est présenté dans les équations (2) à (7). L'objectif consiste à maximiser la somme du prix d'exécution des tâches (2), capturé via une contrainte de collecte de prix (3). Chaque ressource secondaire est une sous-séquence de la ressource globale (4), et effectuer une tâche sur la ressource globale implique de le faire également sur la ressource secondaire associée, et vice-versa (7). Les contraintes temporelles d'exécution sont appliquées via des contraintes de temps de transition (5), (6).

$$\max Z \tag{2}$$

t.q.

$$\texttt{CollecteDePrix}(G, Z, z) \tag{3}$$

$$\texttt{SousSéquence}(G, S_i) \qquad \forall i \in 0..|R| \tag{4}$$

$$\texttt{TempsDeTransition}(G, D, d, O) \tag{5}$$

$$\texttt{TempsDeTransition}(S_i, D, d, Tr) \qquad \forall i \in 0..|R| \tag{6}$$

$$\texttt{LienDeNœud}(G, S_i, t_{ik}) \qquad \forall i \in 0..|R|, \forall t \in T_i \tag{7}$$

3.2 Recherche

La structure de la recherche est différente de celle rencontrée habituellement en PPC lorsque l'on travaille avec des variables de séquence. Elle se décrit généralement via 2 étapes :

Sélection de nœud : un nœud n'étant ni inclus ni exclu dans la séquence est choisi.

Inclusion de nœud : étant donné le nœud choisi à l'étape précédente, des branches considérant de l'insérer, le rendre obligatoire et/ou l'exclure sont générées puis évaluées.

Les branches sont explorées en recherche en profondeur. Pour le STCP-URCPRS, le branchement change dynamiquement en fonction de deux situations :

1. La séquence globale G comporte au moins un nœud v requis mais non-inséré ($v \in R \wedge v \notin \overrightarrow{G}$,

par exemple si un nœud a été inséré dans une ressource secondaire mais plusieurs positions sont possibles pour l'insérer dans la ressource globale). Le branchement suivant est alors effectué :

Sélection de nœud : le nœud requis mais non inséré ayant le minimum de points d'insertions est choisis.

Inclusion du nœud : une branche est générée par lieu d'inclusion du nœud, en privilégiant d'abord les insertions du nœud là où beaucoup d'autres nœuds peuvent également être insérés.

2. Si aucun nœud ne correspond au critère précédent, le branchement s'opère alors comme suit :

Sélection de nœud : le nœud ayant le prix associé le plus grand est choisi. De plus, les nœuds requis mais non insérés sont choisis en priorité.

Inclusion du nœud : une branche est générée par lieu d'inclusion du nœud, en privilégiant d'abord les insertions du nœud où beaucoup d'autres nœuds peuvent également être insérés. Une branche est également générée pour pouvoir exclure le nœud.

Ce branchement est complexe et peut très probablement être amélioré et simplifié. En dépit de sa complexité, les opérations sont relativement rapides, étant donné que récupérer le nombre d'insertions d'un nœud v où le nombre de nœuds pouvant être inséré après un nœud u sont des opérations en temps constants, grâce à des compteurs maintenus dans le domaine [1].

4 Expériences

La recherche exacte employée avec les variables de séquence donne des résultats assez négatifs. Celle-ci est plus lente et moins performante que les autres méthodes proposées [3, 2].

Les variables de séquence étant cependant plus performantes lorsque couplées avec de la Recherche à Large Voisinage (RLV), nous reportons les performances dans ce scénario. La RLV, similaire à celui de [2], consiste à partir de la meilleure solution courante et relaxer (c.à.d. enlever) des tâches successives sur la ressource globale. Ces tâches sont également relaxées sur les ressources secondaires correspondantes. Avec le temps gagné dans la planification suite au retrait de ces tâches, d'autres peuvent être insérées, pour trouver de nouvelles solutions améliorantes. Cette heuristique perd cependant la preuve d'optimalité, la justifiant surtout pour des problèmes de grande taille.

Les résultats sur deux set d'instances sont repris dans la Table 1 et comparés avec ILS, l'état de l'art pour des approches heuristiques. Le délai maximum a été mis à 30 minutes. Pour les instances les plus larges (n plus élevé), les variables de séquence ont le dessus. Ces résultats sont toutefois très préliminaires : 10 itérations sont reportées et agrégées pour ILS, alors qu'une seule est reportée dans l'autre cas.

		Séquence		ILS		
				Écart		
Set	n	Écart	Temps	Moy.	Meil.	Temps
A	50	0.22	128.90	0.00	**0.00**	**21.70**
	100	0.04	123.77	0.00	**0.00**	**67.60**
	150	0.02	**118.21**	0.01	**0.00**	**116.36**
	200	0.08	182.52	0.01	**0.00**	**161.82**
	250	0.01	232.58	0.00	**0.00**	**208.79**
	300	0.05	**257.22**	0.01	**0.00**	**254.50**
	350	0.01	**150.03**	0.01	**0.00**	294.06
	400	0.05	**129.71**	0.01	**0.00**	337.25
	450	0.00	**99.18**	0.01	**0.00**	379.37
	500	0.07	**98.47**	0.02	**0.01**	420.74
	All	0.05	**152.06**	0.01	**0.00**	226.22
T	50	1.75	22.57	0.00	**0.00**	**16.19**
	100	**0.00**	157.53	0.18	**0.00**	**59.80**
	150	**0.00**	183.85	0.48	0.13	**128.88**
	200	**0.00**	**208.59**	0.93	0.47	235.41
	250	**0.00**	**343.58**	0.70	0.29	366.72
	300	**0.00**	**459.16**	0.36	0.01	517.49
	350	**0.00**	**577.17**	0.33	**0.00**	676.91
	400	**0.00**	**665.67**	0.29	**0.00**	859.58
	450	**0.00**	**718.26**	0.28	**0.00**	1065.05
	500	**0.00**	**736.05**	0.28	**0.00**	1282.29
	All	0.18	**407.24**	0.38	**0.09**	520.83

TABLE 1 – Comparaison entre ILS et les Variables de Séquence. L'écart entre la meilleure solution connue et la solution trouvée est reporté en pourcentage. Le temps est mis en secondes. Moy. et Meil. désignent la moyenne et meilleure de 10 exécutions.

5 Conclusion

Le Séquencement des Travaux de Collecte de Prix avec Une Ressource Commune et Plusieurs Ressources Secondaires est un problème de planification dont la résolution est complexe. Les variables de séquence, habituellement employées sur des problèmes de routage de véhicules, permettent une résolution efficace des instances de taille importante, malgré de faibles performances sur des instances de moindre taille. La complexité de la résolution est actuellement fortement présente dans la procédure de branchement, ce que les travaux futurs devraient tenter d'améliorer.

Références

[1] Augustin Delecluse, Pierre Schaus et Pascal Van Hentenryck : Sequence variables for routing problems. *In 28th International Conference on Principles and Practice of Constraint Programming (CP 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[2] Aurélien Froger et Ruslan Sadykov : New exact and heuristic algorithms to solve the prize-collecting job sequencing problem with one common and multiple secondary resources. *European Journal of Operational Research*, 306(1):65–82, 2023.

[3] Matthias Horn, Johannes Maschler, Günther R Raidl et Elina Rönnberg : A*-based construction of decision diagrams for a prize-collecting scheduling problem. *Computers & Operations Research*, 126:105125, 2021.

[4] Matthias Horn, Günther Raidl et Christian Blum : Job sequencing with one common and multiple secondary resources : A problem motivated from particle therapy for cancer treatment. *In Machine Learning, Optimization, and Big Data : Third International Conference, MOD 2017, Volterra, Italy, September 14–17, 2017, Revised Selected Papers 3*, pages 506–518. Springer, 2018.

[5] Charles Thomas, Roger Kameugne et Pierre Schaus : Insertion sequence variables for hybrid routing and scheduling problems. *In International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 457–474. Springer, 2020.