LocalSolver 11.0 : nouveautés et améliorations des performances pour les modèles ensemblistes

Julien Darlay

LocalSolver, 24 avenue Hoche, Paris, France
jdarlay@localsolver.com

Mots-clés : *solveur, recherche locale*

1 Introduction

LocalSolver est un solveur global de type *model & run* [1] basé sur des techniques exactes et heuristiques. Son formalisme d'entrée lui permet d'accepter tout modèle utilisant les opérateurs mathématiques usuels (arithmétiques, logiques, relationnels, etc.) avec des variables continues, entières ou ensemblistes. Les techniques exactes permettent d'obtenir de bornes et de prouver l'optimalité. Les heuristiques permettent de trouver rapidement de bonnes solutions et de passer à l'échelle sur les plus grosses instances.

Le formalisme de modélisation ensembliste de LocalSolver est particulièrement adapté pour la résolution des problèmes d'optimisation de la *Supply Chain* tels que l'optimisation de tournées de véhicules ou l'ordonnancement de tâches. Ce formalisme a été élargi récemment avec l'introduction de nouveaux opérateurs de modélisation. Les techniques de résolution et de calcul de bornes ont également été renforcées sur ces problèmes. Cette présentation fera le tour des nouveautés introduites dans LocalSolver et illustrera les gains de performances sur des instances de la littérature.

2 Modélisation

Une des innovations de LocalSolver est sa capacité à gérer des variables de décision ensemblistes *list* et *set* qui représentent respectivement des ensembles ordonnés et non-ordonnés. Ces variables permettent de modéliser de façon compacte des problèmes de tournées de véhicules, d'ordonnancement ou de *packing*. Dans les premiers cas une variable *list* représente la séquence des clients à visiter par un camion ou la liste des tâches à effectuer sur une machine. Dans le dernier cas une variable *set* représente l'ensemble des objets à mettre dans une boite.

Nous avons introduit l'opérateur ensembliste *cover* en complément des opérateurs *partition* et *disjoint* déjà existants. Ces trois opérateurs permettent d'exprimer que chaque objet doit appartenir à au moins une collection, exactement une collection et au plus une collection. L'opérateur *cover* est particulièrement utile pour modéliser les problèmes de tournées de véhicules où un client peut être livré par plusieurs camions comme dans le *Split Delivery Vehicle Routing Problem* [2].

L'opérateur *find* permet de savoir à quelle collection est affectée un objet. Il a également été introduit afin de simplifier la modélisation des problèmes de *flexible scheduling* [3] et d'*assembly line balancing* [4].

3 Recherche de solutions

Les modèles de *clustering* géométriques sont particulièrement utiles lors de l'optimisation stratégique d'un réseau de distribution ou pour faire un prédécoupage en régions avant de

résoudre un grand problème de tournées de véhicules. LocalSolver tire parti de la structure de ces modèles en exploitant des voisinages de recherche locale dédiés qui permettent de regrouper des points proches et de séparer des points distants. L'ajout de ces voisinages permet au solveur d'obtenir des solutions à moins de 1% de gap des meilleures solutions connues de la littérature sur des instances allant jusqu'à 80 000 points en quelques secondes.

Les problèmes de tournées de véhicules sont des candidats naturels à la modélisation ensembliste. Les dernières versions de LocalSolver ont mis l'accent sur les problèmes avec fenêtres de temps tels que les *CVRPTW* et *PDPTW*. Des nouveaux voisinages à base de *destroy & repair* ainsi que des voisinages permettant de calculer la meilleure réinsertion d'une séquence de clients permettent de réduire le gap au bout de 5 minutes à moins de 5% sur les instances de la littérature du *CVRPTW* et moins de 8% sur les instances de la littérature du *PDPTW*.

La résolution des problèmes d'ordonnancement flexibles a été renforcée par l'ajout d'heuristiques constructives générant de nombreuses solutions. Ces solutions sont améliorées par recherche locale avec des voisinages classiques et une structure de propagation permettant de répercuter des modifications locales d'une solution sur le reste du planning. Ces nouveautés permettent d'obtenir en une minute des solutions à environ 2% des meilleures solutions connues pour le *flexible jobshop scheduling* et permettent d'améliorer les meilleures solutions connues sur les plus grandes instances d'*assembly line balancing*.

4 Calcul de bornes

Le calcul de bornes sur un modèle ensembliste nécessite de reformuler le problème d'origine pour faire appel aux techniques de programmation mathématiques classiques. LocalSolver reformule automatiquement un problème de tournées de véhicules comme un programme linéaire auquel sont ajoutées des coupes d'élimination de sous-tours et de capacité [5] après chaque résolution. Ces bornes combinées aux techniques déjà existantes permettent à LocalSolver de fermer certaines instances et d'afficher des gaps inférieurs à 2% sur des *TSP* jusqu'à 2000 villes et des gaps inférieurs à 5% pour des *CVRP* jusqu'à 100 clients en moins de 5 minutes.

Références

[1] F. Gardi, T. Benoist, J. Darlay, B. Estellon and R. Megel. Mathematical Programming Solver Based on Local Search. Wiley, 2014.

[2] C. Archetti and M.G. Speranza. Vehicle routing problems with split deliveries. Intl. Trans. in Op. Res., 2012

[3] P. Brucker and R. Schlie. Job-shop scheduling with multi-purpose machines. Computing, 45, 1990

[4] A. Scholl. Balancing and sequencing of assembly lines. 2nd ed., Physica, Heidelberg, Chapter 7, 1999

[5] P. Augerat, J.M. Belenguer, E. Benavent, A. Corberán and D. Naddef. Separating capacity constraints in the CVRP using tabu search. European Journal of Operational Research, 106, 1998