POLYPUBLIE

Learning in local branching

Defeng Liu, Andrea Lodi

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

Date	2021
Type	Communication de conférence / Conference or Workshop Item Résumé/Abstract
Version	Version officielle de l'éditeur / Published version
URL de PolyPublie **PolyPublie URL**	https://publications.polymtl.ca/49755/
Évaluation par les pairs **Peer reviewed**	Révisé par les pairs / Refereed
Conditions d'utilisation **Terms of use**	Creative Commons Attribution 4.0 International (CC BY)
Référence **Citation**	Liu, D., & Lodi, A. (octobre 2021). Learning in local branching [Résumé]. 27th International Conference on Principles and Practice of Constraint Programming (CP 2021), En ligne (2 pages). Publié dans Leibniz International Proceedings in Informatics (LIPIcs), 210. https://doi.org/10.4230/lipics.cp.2021.3

Publication officielle
Official publication

Nom de la conférence **Conference name**	27th International Conference on Principles and Practice of Constraint Programming (CP 2021)
Date et lieu **Date and location**	2021-10-25 - 2021-10-29, En ligne
Maison d'édition **Publisher**	Dagstuhl Publishing
URL officielle **Official URL**	https://doi.org/10.4230/lipics.cp.2021.3
Mention légale **Legal notice**	©Defeng Liu and Andrea Lodi; licensed under Creative Commons License CC-BY 4.0

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal
https://publications.polymtl.ca

Learning in Local Branching

Defeng Liu
CERC, Polytechnique Montréal, Canada

Andrea Lodi[1]
Jacobs Technion-Cornell Institute, Cornell Tech and Technion – Israel Institute of Technology, New York, NY, USA
CERC, Polytechnique Montréal, Canada

Abstract

Although state-of-the-art solvers for Mixed-Integer Programming (MIP) experienced a dramatic performance improvement over the past decades, the resolution of some MIPs is still challenging, requiring hours of computations while, in practice, high-quality solutions are often required to be computed within a very restricted time frame. In such cases, it might be preferable to provide *anytime solutions*, i.e., a first reasonable solution should be generated as early as possible, then better ones produced in the subsequent computation with the user deciding where to stop.

In this respect, the *local branching* (LB) heuristic [2] was proposed to improve an incumbent solution either at very early stages of the computation within a general MIP framework or as a stand-alone algorithmic framework. Roughly speaking, given a feasible solution, the method iterates by first defining a solution neighborhood through the so-called *local branching cut*, then by exploring it by calling a black-box MIP solver. In the local branching algorithm, the choice of the neighborhood size is crucial to performance. In principle, it is desirable to have neighborhoods to be relatively small for efficient computation but still large enough to contain improving solutions. In [2], the size of the neighborhood is mostly initialized by a fixed constant value, then adjusted at run time. Nonetheless, it is reasonable to believe that there is no *a priori* single best neighborhood size and the choice of the value should depend on the characteristics of the problem. Furthermore, it is worth noting that, in many applications, instances of the same problem are solved repeatedly. Real-world problems have a rich structure: while more and more data points are collected, patterns and regularities appear. Therefore, problem-specific and task-specific knowledge can be learned from data and applied to adapting the corresponding optimization scenario. This motives a broader paradigm of sizing the solution neighborhoods in local branching.

Following the line of work analyzed and surveyed in [1] on the use of Machine Learning (ML) for combinatorial optimization, in this work, we aim to guide the (local) search of the local branching heuristic by ML techniques. In particular, given a problem instance and a time limit for (heuristically) solving it, we exploit ML tools to predict reasonable good values of the neighborhood size, in order to maximize the performance of the local branching algorithm. We computationally show that the neighborhood size can indeed be learnt leading to improved performances and that the overall algorithm generalizes well both with respect to the instance size and, more surprisingly, across instances.

2012 ACM Subject Classification Computing methodologies → Machine learning; Mathematics of computing → Discrete optimization

Keywords and phrases Local search, learning, mixed-integer programming

Digital Object Identifier 10.4230/LIPIcs.CP.2021.3

Category Invited Talk

Acknowledgements The authors are indebted to Matteo Fischetti for several conversations about the work in pregress.

[1] Corresponding Author and Invited Speaker



© Defeng Liu and Andrea Lodi;
licensed under Creative Commons License CC-BY 4.0



References

1 Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

2 Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical programming*, 98(1-3):23–47, 2003.