Ph.D. proposition

"Using machine learning techniques to accelerate the proof of optimality in branch-and-cut algorithms for the NP-hard combinatorial optimization problem."

It is well known that branch-and-cut algorithms are the most efficient tools for solving NP-hard combinatorial optimization problems to optimality. The most famous example is the Concorde program written by Applegate et al. for solving the Traveling Salesman Problem (TSP). The branch-and-cut algorithm is a variant of the Branch-and-bound algorithm where one may generate valid inequalities at branch-and-bound tree nodes to cut off the current fractional solution. These valid inequalities could be generated either from facet-defining inequalities of the convex hull of the (integer) feasible solutions of the problem or from generic valid inequalities of Mixed Integer Programming (MIP) theory. The first, called facet cuts, often are more efficient than the second, called generic cuts, as they are more specific to the structure of the problem. But generating facet cuts may be very hard as it requires solving the associated separation problem, usually NP-hard itself. Branch-and-cuts algorithms also rely on branching on integer (zero-one) variables to cut off the current fractional solution. All the art of branch-and-cut algorithms consists in

- making a judicious choice of the node in the branch-and-bound tree to process,
- at the chosen node, choosing generating cuts or branching,
- and finally, determining which kind of cuts to generate for the first case? or which variable to branch on for the second case.

Different choices in these steps may lead to very other performances of branch-and-cut algorithms. Therefore, from the early day of branch-and-cut algorithms, many rules of branching (strong branching, …) and generating cuts (deepest cut…) have been introduced in the literature. However, no rule could be shown to be the best as the performance depends heavily on the problem, especially on the instance. Because of this data dependence, recently, many works have proposed machine learning methods to advise the right rules of generating cuts and branching.

However, several pathologies persist and can considerably slow down branch-and-cut algorithms, such as the tailing-off effect. The latter happens when the branch-and-cut algorithm already finds an optimal solution (by heuristic or by solving relaxations) to the problem but has great difficulty proving the solution's optimality.

In many situations, especially for hard instances of combinatorial optimization, the tailing-off effect frequently appears. In this case, the pathology is that branch-and-cut algorithms may spend more than half of the total execution time to close the gap at less than 1% to optimality. The tailing-off effect may be due to two following reasons:

- Necessary time devoted to fin violated cuts at the nodes of the branch-and-bound tree, but their addition to the relaxation improves only very little the lower bound.
- Branching on variables does not allow to fathom nodes in the branch-and-bound tree anymore.

Hence, the tailing-off effect causes severe pathology of branch-and-cuts algorithms when solving hard instances. To our knowledge, there is no known solution to overcome this effect in the literature. The Ph.D. thesis proposes efficient solutions to the tailing-off effect using machine learning techniques. In contrast with the existing learning methods, we apply machine learning methods only when the tailing-off effect appears. The Ph.D. thesis should devise generating cut and branching rules that can help to eliminate the tailing-off effect. These rules should consider the specific structure of the problem by designing efficient rules to generate facet cuts. The demonstration of the efficiency of rules will be through case studies on hard combinatorial optimization problems with high practical impacts, such as TSP and MaxCut.

References:

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

[2] Nina Balcan, Siddharth Prasad, Tuomas Sandholm, Ellen Vitercik. Sample Complexity of Tree Search Configuration: Cutting Planes and Beyond. Conference on Neural Information Processing Systems, 2021.

[3] Nina Balcan, Travis Dick, Tuomas Sandholm, and Ellen Vitercik. Learning to Branch. International Conference on Machine Learning, 2018.