

A Local Branching Heuristic for MINLPs

GIACOMO NANNICINI¹, PIETRO BELOTTI², LEO LIBERTI¹

Local branching [2] was introduced as a primal heuristic for Mixed-Integer Linear Programs (MILPs) within the context of a Branch-and-Bound (BB) algorithm. The natural setting for local branching is within problems with binary variables. The idea is as follows: whenever a new incumbent is found, a new problem is solved, with the same constraints and objective function as the original problem, but with the addition of a local branching constraint, whose purpose is to allow only a given number of binary variables to change their value with respect to the incumbent. This new problem defines a neighbourhood which is explored by employing a BB algorithm. The computational time required to solve the local branching problem is typically very small. Computational experiments have shown that this simple idea is often able to improve the incumbent on a large number of real-world test problems. Clearly, this paradigm can be directly applied to BB methods for nonconvex Mixed-Integer Nonlinear Programs (MINLPs): the only difference is that the local branching problem should be solved by employing a BB algorithm for MINLPs. However, branch-and-bound methods for this class of problems are in practice significantly slower than in the linear case, because branching can occur on integer and continuous variables [1], a convexification refinement step is applied, and large continuous Nonlinear Programs (NLPs) are solved at some nodes. Therefore, the exploration of the neighbourhood may require significantly more time, and the heuristic becomes less appealing.

In this paper, we propose a local branching scheme for nonconvex MINLPs, which is based on repeatedly solving a sequence of limited-size MILPs and NLPs. Each of these problems has a different purpose. We use a NLP to estimate the descent direction of the original (nonlinear) objective function. Then, we solve a MILP on the convexification of the feasible region with a local branching constraint, to enforce integral feasibility while trying to follow the descent direction estimated at the previous step. Finally, we fix the integer variables and try to satisfy the original constraints via a NLP. In case of failure, we cut off the computed solution from the MILP, and iterate the algorithm. As each solved problem has a smaller feasible region or involves fewer variables with respect to the original MINLP, this approach is fast. Preliminary computational experiments, run with Cplex 11.0 and `ipopt` with an AMPL script, confirm its usefulness. We considered 21 convex and nonconvex instances taken from MINLPLib, and obtained initial feasible points from `couenne` [1]. For 30 out of 39 initial points (76.9%), our heuristic was able to find a better solution. In 24 cases (61.5%), an improved incumbent is found at the first iteration. The improvement is significant. On 4 instances, our approach returns the best known solution (reported on the MINLPLib website) from the first feasible solution found by `couenne`. On an additional instance, the best known solution is returned from the second feasible solution found by `couenne`.

References

- [1] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. Technical Report RC24620, IBM, 2008.
- [2] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–37, 2005.

¹LIX, École Polytechnique, F-91128 Palaiseau, France; {giacomon, liberti}@lix.polytechnique.fr
²Lehigh University, 200 West Packer Avenue, Bethlehem, PA 18015; belotti@lehigh.edu