A Multi-Reference Relaxation Enforced Neighborhood Search Heuristic in SCIP

Suresh Bolusani*[1][0000-0002-5735-3443], Gioni Mexi[1][0000-0003-0964-9802], Mathieu Besançon[2][0000-0002-6284-3033], and Mark Turner[1][0000-0001-7270-1496]

[1] Zuse Institute Berlin, Germany
[2] Grenoble Alpes University, Inria, LIG
{bolusani, mexi, turner}@zib.de, mathieu.besancon@inria.fr

Abstract. This paper proposes and evaluates a Multi-Reference Relaxation Enforced Neighborhood Search (MRENS) heuristic within the SCIP solver. This study marks the first integration and evaluation of MRENS in a full-fledged MILP solver, specifically coupled with the recently-introduced Lagromory separator for generating multiple reference solutions. Computational experiments on the MIPLIB 2017 benchmark set show that MRENS, with multiple reference solutions, improves the solver's ability to find higher-quality feasible solutions compared to single-reference approaches. This study highlights the potential of multi-reference heuristics in enhancing primal heuristics in MILP solvers.

Keywords: Mixed-integer optimization · Heuristics

1 Introduction

Primal heuristics aim to find feasible solutions to optimization problems at a lower computational cost than their exact counterparts but without any guarantees. For mixed-integer linear programming (MILP), two main use cases spur the development of primal heuristics. First, as standalone methods, heuristics allow practitioners to obtain feasible solutions in shorter amounts of time. Second, as components of exact solution algorithms, heuristics provide feasible solutions for various purposes in these algorithms, e.g., for pruning nodes in a branch-and-cut algorithm.

In this work, we consider general MILP problems, which take the form:

$$\min_x \{c^\intercal x \mid Ax \geqslant b, \;\; l \leqslant x \leqslant u, \;\; x \in \mathbb{Z}^{|\mathcal{I}|} \times \mathbb{R}^{n-|\mathcal{I}|}\}. \tag{1}$$

A variety of heuristics exist in the MILP solvers, e.g., rounding, diving, objective diving, and improvement heuristics. Interested readers may refer to [3, 1] for an overview. Most of these heuristics require a fractional solution as a reference solution, and a few of them also require an incumbent feasible solution. The fractional solution is often obtained from the linear programming (LP)

* Corresponding author.

relaxation of (1), which is defined as follows:

$$\min_x \{c^\mathsf{T} x \mid Ax \geqslant b,\ \ l \leqslant x \leqslant u,\ \ x \in \mathbb{R}^n\}. \tag{2}$$

The usage of this fractional solution varies depending on the heuristic, e.g., *relaxation enforced neighborhood search* (RENS) [3, 1, 4] and *feasibility pump* (FP) [7, 2, 8]. Despite the existence of a wide variety of heuristics, there is one common aspect among them, i.e., all of them require and use a *single* reference solution for their execution.

The literature for MILP heuristics exploiting *multiple* reference solutions is sparse. To the best of our knowledge, the only such work is [12] in which the authors improved the FP heuristic by considering multiple reference solutions. They proposed a new heuristic called mRENS and programmed it in a standalone implementation of the FP heuristic. They observed performance improvements on various MILP test sets. However, the idea was never tested and integrated into a full-fledged MILP solver.

The lack of literature using multiple reference solutions in a heuristic setting is also reflected in the optimization software landscape, where we are aware of no open-source implementations of such heuristics in an MILP solver. Motivated by this, along with the recent integration of a new relax-and-cut framework-based separator (cutting plane generation routine) in SCIP (more details in Section 2), we implement a new primal heuristic that considers multiple reference solutions within the state-of-the-art solver SCIP [5]. We call our new heuristic MRENS, similar to the heuristic in [12].

The differences between our proposed heuristic and the mRENS of [12] are threefold. First, we consider the multiple fractional solutions generated as a byproduct of the recently integrated *Lagromory* separator [5] in SCIP, whereas [12] considers the multiple solutions generated in the pumping loop of the FP heuristic. Second, the sub-MILPs of our heuristics differ in their feasible regions. Finally, we use the working limits of the RENS heuristic in SCIP [1], which are different from those of [12].

In the rest of this paper, we present the details of our heuristic in Section 2, our computational experiments and analysis in Section 3, and our conclusions in Section 4.

2 Multi-Reference Relaxation Enforced Neighborhood Search (MRENS)

RENS [3, 1, 4] is a rounding primal heuristic for MILPs that searches for an (integral) primal feasible solution of (1) in the neighborhood of a given single fractional solution, e.g., an optimal solution of (2). MRENS, *multi-reference relaxation enforced neighborhood search*, is a generalization of RENS in the sense that MRENS considers multiple fractional solutions to define the neighborhood. We will now describe MRENS in Section 2.1 and the generation procedure of multiple reference solutions in Section 2.2.

2.1 MRENS

Let $\bar{x}^{(0)}$ be a fractional optimal solution of (2). A trivial rounding heuristic applied at $\bar{x}^{(0)}$ may fail to find a primal feasible solution of (1) most of the time because it ignores the linear constraints $Ax \geqslant b$. RENS instead solves the sub-MILP (3) and finds the best possible rounding of $\bar{x}^{(0)}$ in the feasible region of (1).

$$\min_x \{c^\mathsf{T} x \mid Ax \geqslant b,\ l \leqslant x \leqslant u,\ \lfloor \bar{x}_j^{(0)} \rfloor \leqslant x_j \leqslant \lceil \bar{x}_j^{(0)} \rceil\ \forall j \in \mathcal{I},\ x \in \mathbb{Z}^{|\mathcal{I}|} \times \mathbb{R}^{n-|\mathcal{I}|}\}. \tag{3}$$

While constructing (3), the integer variables with integral values in $\bar{x}^{(0)}$ are fixed to these values, and the domains of remaining integer variables are changed based on their fractional values in $\bar{x}^{(0)}$. Consequently, the sub-MILP (3) can be as computationally hard as the original MILP (1), contingent on the variable fixings and domain changes. For example, if the original problem is a pure binary MILP, the sub-MILP constructed with a fully fractional reference solution results in the same original problem. In practice, however, difficult sub-MILPs occur infrequently, and it has been observed empirically [4] that RENS typically produces over-restricted and thus infeasible sub-MILPs.

MRENS, by design, aims to overcome the observed issues of over-restricted sub-MILPs through the use of multiple reference solutions $\{\bar{x}^{(0)}, \bar{x}^{(1)}, \bar{x}^{(2)}, \ldots, \bar{x}^{(k)}\}$. Let $\mathcal{J} = \{0, 1, 2, \ldots, k\}$, $x_{j\,\min} = \min_{i \in \mathcal{J}} \bar{x}_j^{(i)}$, and $x_{j\,\max} = \max_{i \in \mathcal{J}} \bar{x}_j^{(i)}$. Then, in MRENS, we solve the sub-MILP (4) instead of the sub-MILP (3).

$$\begin{aligned} \min_x \{c^\mathsf{T} x \mid Ax \geqslant b,\ l \leqslant x \leqslant u,\ x \in \mathbb{Z}^{|\mathcal{I}|} \times \mathbb{R}^{n-|\mathcal{I}|},& \\ \lceil x_{j\,\min} \rceil \leqslant x_j \leqslant \lfloor x_{j\,\max} \rfloor\ \forall j \in \mathcal{I}\ \text{ if } x_{j\,\max} - x_{j\,\min} \geqslant 1.0,& \\ \lfloor x_{j\,\min} \rfloor \leqslant x_j \leqslant \lceil x_{j\,\max} \rceil\ \forall j \in \mathcal{I}\ \text{ if } x_{j\,\max} - x_{j\,\min} < 1.0\}.& \end{aligned} \tag{4}$$

The sub-MILPs (4) and (3) are equivalent if we consider only the single fractional solution $\bar{x}^{(0)}$, and otherwise, (4) is a relaxation of (3). Accordingly, (4) may be computationally more difficult to solve than (3) but also has a better chance of producing primal feasible solutions of (1).

2.2 Generation of Multiple Reference Solutions

Recently, the Lagromory separator based on the (iterative) relax-and-cut framework [9] was implemented in SCIP 9.0 [5]. The following steps describe iteration k of this separator call. For $k = 0$, the LP in steps 2 and 3 refers to (2) and $\bar{x}^{(0)}$ is its fractional optimal solution.

1. Generate Gomory mixed-integer (GMI) cuts that separate $\bar{x}^{(k)}$.
2. Add the cuts to the objective function of the LP in a Lagrangian fashion.
3. Update the Lagrangian multipliers and the objective function of the LP.
4. Solve this new LP to obtain its optimal solution $\bar{x}^{(k+1)}$.
5. If integral, save $\bar{x}^{(k+1)}$ as a feasible solution to (1). Otherwise, set $k = k + 1$ and go to step 1.

Multiple such iterations are performed in a single call to the separator while keeping the feasible region of (2) intact. The outcome is a set of GMI cuts from multiple LP feasible bases of (2). Since it solves multiple LPs, this separator is computationally costly compared to other separators present in SCIP. So, currently, this separator does not improve the default SCIP and is inactive. The lack of improvement motivated the search for ways to exploit different information generated throughout the separation algorithm, particularly on the primal side. Thus, we propose to use these bases as the multiple reference solutions.

Using too many reference solutions can cause the sub-MILP to have a large feasible region, inducing a prohibitive solving time. Therefore, we consider a maximum of three reference solutions, which follows the empirical results of [12]. Specifically, we consider $\left\{\bar{x}^{(0)}, \bar{x}^{(M-1)}, \bar{x}^{(M)}\right\}$ if the separator generates $M \geqslant 2$ solutions. Considering $\bar{x}^{(0)}$ ensures that the MRENS subproblem is a relaxation of the RENS subproblem, which is our design choice. As for $\bar{x}^{(M-1)}$ and $\bar{x}^{(M)}$, note that executing the separator without working limits results in the rank-1 GMI closure. So, intuitively, the latter solutions in the generated solutions' sequence are closer to the MILP primal feasible solutions. Hence, we consider $\bar{x}^{(M-1)}$ and $\bar{x}^{(M)}$, along with $\bar{x}^{(0)}$, as the reference solutions for MRENS.

3 Computational Results

We implemented and tested our heuristic in SCIP 9.0.0[1]. We did our experiments on a cluster equipped with Intel Xeon Gold 5122 CPUs and a limit of 96GB of RAM. We used the MIPLIB 2017 benchmark library [10] as our test set. To mitigate the effects of performance variability [11], we solved each problem with five different seeds, namely $\{0, 1, 2, 3, 4\}$, resulting in a total of 1200 problem-seed combinations, which we refer to as instances. SCIP aborts and fails to solve eight instances, so we do not consider them.

We solved each instance with a time limit of two hours and a memory limit of 50 GB. We use the same working limits as RENS in SCIP, i.e., 5000 solving node limit, 500 stalling node limit (the maximum number of nodes an MILP solver can process without improving the incumbent solution of the sub-MILP), and calling the heuristic only in the root node. We also require a minimum of 50% integer variable fixings for heuristic execution while creating the sub-MILP and a minimum of 25% total variable fixings after presolving the sub-MILP.

Recall that the main motivation for this work was to have a heuristic that can use multiple reference solutions from the Lagromory separator. Accordingly, we activate this separator but use it as a routine for generating reference solutions only, i.e., the generated GMI cuts are not added to SCIP. Furthermore, we deactivate the RENS heuristic to avoid redundant computational costs.

In the following, we compare two settings, "MRENS-L" and "RENS-L" (L for Lagromory). In the MRENS-L setting, we use three reference solutions as detailed in Section 2.2. In the RENS-L setting, we use only the first solution $\bar{x}^{(0)}$,

[1] SCIP commit hash: dadaf6a544b39ee20a64d1dde942b2b8b1164b7e

which is an optimal solution of (2). So, the MRENS heuristic in the RENS-L setting and RENS in SCIP are almost identical except for their execution frequency. The former requires that the Lagromory separator generate a fractional solution, whereas the latter does not have such a requirement and can utilize the node relaxation's fractional solution directly.

Table 1 provides a comparison of the two settings. As expected, MRENS-L fixes fewer integer variables while constructing its sub-MILP (4) compared to RENS-L. Accordingly, MRENS-L is executed less often than RENS-L because of the requirement of minimum 50% integer variable fixings. MRENS-L is also more successful in finding feasible solutions than RENS-L. More importantly, MRENS-L finds the best-known solution more frequently than RENS-L, with success rates of 17.0% and 12.3%, respectively.

Table 1. Aggregated results comparing MRENS-L and RENS-L. The columns in order of appearance: the heuristic setting used; the total number of heuristic calls over all instances; the percentages of heuristic calls where the heuristic was executed (i.e., the calls where at least 50% of the integer variables were fixed in the sub-MILP), successfully found a solution, and found a new best solution; and the percentage of fixed integer variables in the sub-MILP averaged over all heuristic calls.

setting	# calls	% of calls heur. executed	% of calls solution found	% of calls best found	avg. % of fixed int. variables
MRENS-L	1883	78.5%	30.5%	17.0%	73%
RENS-L	1848	83.0%	24.1%	12.3%	77%

In Table 2, we compare the overall performance of SCIP with MRENS-L and RENS-L settings. Both settings solve almost the same number of instances within the time limit. For the instances that were solved by both settings, MRENS-L is faster by 5% and generates branch-and-bound trees with 3% fewer nodes. On the *affected* instances where MRENS-L has impacted the solving process, MRENS-L outperforms RENS-L significantly both in terms of solution time and tree size. Specifically, MRENS-L is 23% faster and generates trees with 15% fewer nodes.

Additionally, both settings are fast within the working limits; on average, they require less than one second of execution time per instance without accounting for the generation time of reference solutions. For the instances that were solved to optimality, MRENS-L found an optimal solution for 38 out of 616, whereas RENS-L found an optimal solution for 13 out of 617. For the instances that were not solved to optimality, MRENS-L found the best solution for 24 out of 562, whereas RENS-L found the best solution for 15 out of 559.

4 Conclusion

Our results demonstrate that the MRENS-L setting finds both feasible and best-known solutions for MILPs more often than the RENS-L setting without much

Table 2. Aggregated results comparing SCIP's performance with MRENS-L and RENS-L settings for the categories: all instances ("all"), instances solved by both settings ("both-solved"), instances where the solving path is different ("affected"), and affected instances solved by both settings ("affected-solved"). The columns refer to the number of instances in each category, the number of solved instances for each setting, the geometric mean of the runtime and nodes for each setting, and the relative quotients of those.

subset	instances	MRENS-L			RENS-L			relative	
		solved	time	nodes	solved	time	nodes	time	nodes
all	1192	632	1136	-	633	1168	-	0.97	-
both-solved	625	625	213	2608	625	224	2698	0.95	0.97
affected	142	134	322	-	135	411	-	0.78	-
affected-solved	127	127	228	5500	127	297	6475	0.77	0.85

additional computational cost, indicating that using multiple reference solutions in the RENS framework is beneficial. The concept may be extended to other heuristics, such as RINS [6], or with other sources of reference solutions.

Acknowledgements

The work for this article has been conducted in the Research Campus MODAL funded by the German Federal Ministry of Education and Research (BMBF) (grant number 05M14ZAM).

References

1. Achterberg, T.: Constraint integer programming. Ph.D. thesis, TU Berlin (2007)
2. Achterberg, T., Berthold, T.: Improving the feasibility pump. Discrete Optimization **4**(1), 77–86 (2007)
3. Berthold, T.: Primal heuristics for mixed integer programs. Master's thesis, Technische Universität Berlin (2006)
4. Berthold, T.: RENS: the optimal rounding. Mathematical Programming Computation **6**, 33–54 (2014)
5. Bolusani, S., Besançon, M., Bestuzheva, K., Chmiela, A., Dionísio, J., Donkiewicz, T., van Doornmalen, J., Eifler, L., Ghannam, M., Gleixner, A., et al.: The SCIP optimization suite 9.0. arXiv preprint arXiv:2402.17702 (2024)
6. Danna, E., Rothberg, E., Pape, C.L.: Exploring relaxation induced neighborhoods to improve MIP solutions. Mathematical Programming **102**, 71–90 (2005)
7. Fischetti, M., Glover, F., Lodi, A.: The feasibility pump. Mathematical Programming **104**(1), 91–104 (2005)
8. Fischetti, M., Salvagnin, D.: Feasibility pump 2.0. Mathematical Programming Computation **1**(2), 201–222 (2009)
9. Fischetti, M., Salvagnin, D.: A relax-and-cut framework for Gomory mixed-integer cuts. Mathematical Programming Computation **3**, 79–102 (2011)

10. Gleixner, A., Hendel, G., Gamrath, G., Achterberg, T., Bastubbe, M., Berthold, T., Christophel, P., Jarck, K., Koch, T., Linderoth, J., et al.: MIPLIB 2017: data-driven compilation of the 6th mixed-integer programming library. Mathematical Programming Computation pp. 1–48 (2021). https://doi.org/10.1007/s12532-020-00194-3
11. Lodi, A., Tramontani, A.: Performance variability in mixed-integer programming. In: Theory Driven by Influential Applications, pp. 1–12. INFORMS (2013). https://doi.org/10.1287/educ.2013.0112
12. Mexi, G., Berthold, T., Salvagnin, D.: Using multiple reference vectors and objective scaling in the feasibility pump. EURO Journal on Computational Optimization **11**, 100066 (2023)