Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

TIMO BERTHOLD

RENS
the optimal rounding

Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

ZIB-Report 12-17 (April 2012) revised version February 2013

Herausgegeben vom
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7
D-14195 Berlin-Dahlem

Telefon: 030-84185-0
Telefax: 030-84185-125

e-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

RENS

the optimal rounding

Timo Berthold*

14/Feb/2013♡
(revised version)

Abstract

This article introduces RENS, the *relaxation enforced neighborhood search*, a large neighborhood search algorithm for mixed integer nonlinear programs (MINLPs). It uses a sub-MINLP to explore the set of feasible roundings of an optimal solution $\bar{x}$ of a linear or nonlinear relaxation. The sub-MINLP is constructed by fixing integer variables x_j with $\bar{x}_j \in \mathbb{Z}$ and bounding the remaining integer variables to $x_j \in \{\lfloor \bar{x}_j \rfloor, \lceil \bar{x}_j \rceil\}$. We describe two different applications of RENS: as a standalone algorithm to compute an optimal rounding of the given starting solution and as a primal heuristic inside a complete MINLP solver.

We use the former to compare different kinds of relaxations and the impact of cutting planes on the so-called *roundability* of the corresponding optimal solutions. We further utilize RENS to analyze the performance of three rounding heuristics implemented in the branch-cut-and-price framework SCIP. Finally, we study the impact of RENS when it is applied as a primal heuristic inside SCIP.

All experiments were performed on three publically available test sets of mixed integer linear programs (MIPs), mixed integer quadratically constrained programs (MIQCPs), and MINLPs, using solely software which is available in source code.

It turns out that for these problem classes 60% to 70% of the instances have *roundable* relaxation optima and that the success rate of RENS does not depend on the percentage of fractional variables. Last but not least, RENS applied as primal heuristic complements nicely with existing root node heuristics in SCIP and improves the overall performance.

Keywords: mixed integer programming, mixed integer nonlinear programming, primal heuristic, large neighborhood search, rounding

Mathematics Subject Classification: 90C11, 90C20, 90C30, 90C59

1 Introduction

Primal heuristics are algorithms that try to find feasible solutions of good quality for a given optimization problem within a reasonably short amount of time. There is typically no guarantee that they will find any solution, let alone an optimal one.

For mixed integer linear programs (MIPs) it is well known that general-purpose primal heuristics like the Feasibility Pump [3, 29, 32] are able to find high-quality solutions for a wide range of problems. Over time, primal heuristics have become a substantial ingredient of state-of-the-art MIP solvers [10, 19]. Discovering good feasible solutions at an early stage of the MIP solving process has several advantages:

*Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany, berthold@zib.de

- The bounding step of the branch-and-bound [41] algorithm depends on the quality of the incumbent solution; a better primal bound leads to more nodes being pruned and hence to smaller search trees.
- The same holds for certain presolving and domain propagation strategies such as reduced cost fixing. Better solutions can lead to tighter domain reductions, in particular more variable fixings. This, as a consequence, might lead to better dual bounds and the generation of stronger cutting planes.
- In practice, it is often sufficient to compute a heuristic solution whose objective value is within a certain quality threshold. For hard MIPs that cannot be solved to optimality within a reasonable amount of time, it might still be possible to generate good primal solutions quickly.
- Improvement heuristics such as RINS [28] or Local Branching [30] need a feasible solution as starting point.

Similar statements hold for other classes of mathematical programs. Often, techniques such as reduced cost fixing or cutting planes are more heavily or even exclusively applied at the root node of a branch-and-bound search tree. Therefore, already knowing good solutions during root node processing is significantly more beneficial than finding them later during tree search.

The last fifteen years have seen several publications on general-purpose heuristics for MIPs, including [3, 6, 7, 9, 11, 12, 32, 33, 36, 38, 43, 44, 48, 53]. For an overview, see [10, 34, 35]. For mixed integer nonlinear programming, the last three years have shown a rising interest in the research community for general-purpose primal heuristics [13, 14, 16, 21, 22, 25, 26, 42, 46, 47].

A *mixed integer nonlinear program (MINLP)* is an optimization problem of the form

$$\begin{array}{lll} \min & d^T x & \\ \text{s.t.} & g_i(x) \leq 0 & \text{for all } i \in \mathcal{M} \\ & L_j \leq x_j \leq U_j & \text{for all } j \in \mathcal{N} \\ & x_j \in \mathbb{Z} & \text{for all } j \in \mathcal{I}, \end{array} \tag{1}$$

where $\mathcal{I} \subseteq \mathcal{N} \coloneqq \{1, \ldots, n\}$ is the index set of the integer variables, $d \in \mathbb{R}^n$, $g_i : \mathbb{R}^n \to \mathbb{R}$ for $i \in \mathcal{M} \coloneqq \{1, \ldots, m\}$, and $L \in (\mathbb{R} \cup \{-\infty\})^n$, $U \in (\mathbb{R} \cup \{+\infty\})^n$ are lower and upper bounds on the variables, respectively. Note that a nonlinear objective function can always be reformulated by introducing one additional variable and constraint, hence form (1) is general. We assume without loss of generality that $L_j \leq U_j$ for all $j \in \mathcal{N}$ and $L_j, U_j \in \mathbb{Z}$ for all $j \in \mathcal{I}$.

There are many subclasses of MINLP, the following four will be considered in this article:

- If all constraint functions g_i are quadratic, problem (1) is called a *mixed integer quadratically constrained program (MIQCP)*.
- If all constraint functions g_i are linear, problem (1) is called a *mixed integer program (MIP)*.
- If $\mathcal{I} = \varnothing$, problem (1) is called a *nonlinear program (NLP)*.
- If $\mathcal{I} = \varnothing$ and all g_i are linear, problem (1) is called a *linear program (LP)*.

With a slight abuse of notation, we will use the abbreviation LP for the term linear programming as well as for the term linear program. The same holds for MINLP, MIQCP, MIP, and NLP.

At the heart of many MIP improvement heuristics, such as Local Branching [30], RINS [28], and DINS [33], lies *large neighborhood search (LNS)*, the paradigm of solving a small sub-MIP which promises to contain good solutions. Recently, those LNS improvement heuristics have been extended to the more general case of MINLP [16, 22, 47]. In contrast, Undercover [13, 14] is an LNS start heuristic for MINLP that does not have an equivalent in MIP.

In this paper, we introduce the *relaxation enforced neighborhood search* (RENS), a large neighborhood search algorithm for MINLP. It constructs a sub-MINLP of a given MINLP based on the solution of a relaxation. RENS is designed to compute the optimal – w.r.t. the original objective function – rounding of a relaxation solution.

Many LNS heuristics, diving and of course all rounding heuristics are based on the idea of fixing some of the variables that take an integer value in the relaxation solution. Therefore, the question of whether a given solution of a relaxation is *roundable*, i.e., all fractional variables can be shifted to integer values without losing feasibility for the constraint functions, is particularly important for the likelihood of other primal heuristics to succeed.

We use RENS to analyze the roundability of instances from different classes of mathematical programs, demonstrate the computational impact of using different relaxations, and use these results to evaluate the performance of several rounding heuristics from the literature. Finally, we investigate the effectiveness of RENS applied as a start heuristic at the root node of a branch-and-cut solver. For these experiments, we use general, publically available MIP, MIQCP and MINLP test sets obtained from the MIPLIB 3.0 [18], the MIPLIB 2003 [4], the MIPLIB 2010 [40], the MINLPLib [23] and the MIQCP test set compiled in [15].

The remainder of the paper is organized as follows. Section 2 introduces the generic scheme of the RENS algorithm. In Section 3, we discuss the algorithmic design and describe implementation details, in particular for the application of RENS as a subsidiary method inside a complete solver. The setup for the computational experiments is presented in Section 4. Section 5 provides detailed computational results and Section 6 contains our conclusions.

2 A scheme for an LNS rounding heuristic

Given a mixed integer program, the paradigm of fixing a subset of the variables in order to obtain subproblems that are easier to solve has proven successful in many MIP improvement heuristics such as RINS [28], DINS [33], Mutation, and Crossover [10, 48]. These strategies can be directly extended to MINLP, see [16].

For a given MINLP, the NLP which arises by omitting the integrality constraints ($x_j \in \mathbb{Z}$ for all $j \in \mathcal{I}$) is called the *NLP relaxation* of the MINLP. The LP relaxation of a MIP is defined analogously. For a point $\bar{x} \in [L, U]$ (i.e., $L_j \leq \bar{x}_j \leq U_j$ for all $j \in \mathcal{N}$) the set of all *fractional variables* is defined as $\mathcal{F} := \{j \in \mathcal{I} \mid \bar{x}_j \notin \mathbb{Z}\}$.

Before we formulate the RENS algorithm, let us formalize the notion of an (optimal) rounding:

Definition 2.1 (rounding)**.** *Let $\bar{x} \in [L, U]$. The set*

$$\mathcal{R}(\bar{x}) := \{x \in \mathbb{R}^n \mid x_j \in \{\lfloor \bar{x}_j \rfloor, \lceil \bar{x}_j \rceil\} \text{ for all } j \in \mathcal{I}, L_j \leq x_j \leq U_j \text{ for all } j \in \mathcal{N}\}$$

is called the set of roundings *of $\bar{x}$.*

In general, $\mathcal{R}(\bar{x})$ is a mixed integer set, a disjoint union of $2^{|\mathcal{F}|}$ polyhedra. Note that in the special case of $\mathcal{I} = \mathcal{N}$, so-called pure integer problems, the set of roundings of $\bar{x}$ is a $2^{|\mathcal{F}|}$-elementary lattice, the vertices of an $|\mathcal{F}|$-dimensional unit hypercube:

$$\mathcal{R}(\bar{x}) = \Big\{x \in \mathbb{Z}^n \mid x_{\mathcal{I}\setminus\mathcal{F}} = \bar{x}_{\mathcal{I}\setminus\mathcal{F}}, x_{\mathcal{F}} \in \bigtimes_{j\in\mathcal{F}} \{\lfloor \bar{x}_j \rfloor, \lceil \bar{x}_j \rceil\}\Big\} \subseteq \bigtimes_{j\in\mathcal{I}} \{L_j, \ldots, U_j\}.$$

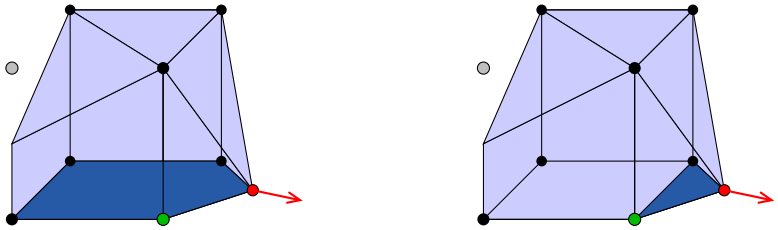
Here, $x_{\mathcal{F}}$ and $x_{\mathcal{I}\setminus\mathcal{F}}$ denote the projection of x to the space of fractional and integral variables, respectively.

Definition 2.2 (optimal rounding)**.** *Let $\bar{x} \in [L, U]$ and $\tilde{x} \in \mathcal{R}(\bar{x})$.*

1. *We call $\tilde{x}$ a* feasible rounding *of $\bar{x}$, if $g_i(\tilde{x}) \leq 0$ for all constraints $i \in \mathcal{M}$ of MINLP* (1)*.*
2. *We call $\tilde{x}$ an* optimal rounding *of $\bar{x}$, if $\tilde{x} \in$ argmin$\{d^T x \mid x \in \mathcal{R}(x), g_i(x) \leq 0$ for all $i \in \mathcal{M}\}$.*
3. *We call $\bar{x}$* roundable *if it has a feasible rounding.*

Because $\mathcal{R}(\bar{x})$ is bounded, $\bar{x}$ either has an optimal rounding or is not roundable.

Figure 1: RENS for MIP: original MIP (light), sub-MIP received by fixing (dark, left) and 0-1 sub-MIP by additional bound reduction (dark, right)



The idea of our newly proposed LNS algorithm is to define a sub-MINLP that optimizes over the set of roundings of a relaxation optimum $\bar{x}$. This is done by fixing all integer variables that take an integral value in $\bar{x}$. For the remaining integer variables, the bounds get tightened to the two nearest integral values, see Figure 1. Note that in the case of a completely fractional relaxation solution to a problem where all integer variables are *binary*, the subproblem would be identical to the original. We will therefore use a threshold for the percentage of integral variables, see next section.

If the sub-MINLP is solved by using a linear outer approximation, tightening the variable bounds to the nearest integers often improves the dual bound, since reduced domains give rise to a tighter linear relaxation. Technically, all integer variables with tightened bounds can be easily transformed to binary variables, by substituting $x'_j = x_j - L_j$. Binary variables are preferable over general integers since many MIP-solving techniques such as probing [49], knapsack cover cuts [5, 37, 54], or the OCTANE heuristic [6] are only used for binary variables.

As the sub-MINLP is completely defined by the relaxation solution $\bar{x}$, we call the procedure *relaxation enforced neighborhood search*, or shortly RENS. Figure 2 shows the basic algorithm, which by construction has some important properties:

Lemma 2.3. *Let the starting point $\bar{x}$ be feasible for the NLP relaxation.*

1. *A point $\tilde{x}$ is a feasible solution of the sub-MINLP if and only if it is a feasible rounding of $\bar{x}$, in particular:*
2. *the optimum of the sub-MINLP is the optimal rounding of $\bar{x}$, and*
3. *if the sub-MINLP is infeasible, then no feasible rounding of $\bar{x}$ exists.*

Two major features distinguish RENS from other MIP and MINLP primal heuristics known from the literature. Firstly, the RENS algorithm does not require a known feasible solution,

Figure 2: Generic RENS algorithm

Input: MINLP P as in (1)
Output: feasible solution $\tilde{x}$ for P or $\emptyset$

1 **begin**
  `/* compute optimal solution of the NLP relaxation of` P `*/`
2 $\bar{x} \leftarrow \text{argmin}\{d^T x \mid g_i(x) \leq 0 \text{ for all } i \in \mathcal{M}, x \in [L, U]\}$;
3 **forall** $j \in \mathcal{I}$ **do**
4 **if** $\bar{x}_j \in \mathbb{Z}$ **then**
5 fix x_j: $L_j \leftarrow \bar{x}_j, U_j \leftarrow \bar{x}_j$;
6 **else**
7 change to binary bounds: $L_j \leftarrow \lfloor \bar{x}_j \rfloor$, $U_j \leftarrow \lceil \bar{x}_j \rceil$;
8
  `/* solve the resulting sub-MINLP of` P `*/`
9 $\tilde{x} \leftarrow \text{argmin}\{d^T x \mid g_i(x) \leq 0 \text{ for all } i \in \mathcal{M}, x \in [L, U], x_j \in \mathbb{Z} \text{ for all } j \in \mathcal{I}\}$;
10 **return** $\tilde{x}$;
11 **end**

unlike other large neighborhood search heuristics that have been described for MIP, namely RINS [28], Local Branching [30], Crossover [10, 48], DINS [33], or Proximity Search [31]. It is a start heuristic, not an improvement heuristic. The same holds for nonlinear variants of these heuristics [16, 22, 47].

Secondly, RENS solves a *single* sub-*MINLP*. In contrast, most primal heuristics for MINLP, in particular the various nonlinear feasibility pump versions [21, 22, 25, 26], RECIPE [42] and Iterative Rounding [46], solve a *series* of auxiliary *MIPs*, often alternated with a sequence of NLPs, to produce a feasible start solution. The number of iterations is typically not fixed, but depends on the instance at hand.

3 Design and implementation details

In this section, we discuss the details of our RENS implementation. A particular focus is set on the application of RENS as a subsidiary method inside a complete branch-and-bound solver.

In principle, an arbitrary point may be used as starting point in line 2 of the RENS algorithm, see Figure 2. Most complete solvers for MINLP are based on branch-and-bound and involve the solution of an NLP relaxation or a linear outer approximation. Their optima are natural choices as starting points. While the NLP optimum is supposed to be "closer" to the feasible region of the MINLP, the LP can usually be computed faster and often gives rise to smaller subproblems. More precisely, the NLP fulfills all nonlinear constraints $g_i(x) \leq 0$, whereas the LP, if solved with the simplex algorithm, tends to fulfill more integrality constraints, which reduces the computational complexity of the RENS subproblem. Thus, both relaxations have their pros and cons; which one proves better in practice will be investigated in our empirical studies, see Section 5.

When using a linear outer approximation (the LP relaxation in case of MIP), an important question is whether we should use the optimum of the initial LP relaxation or the LP solution after cutting planes have been applied. As before, cutting planes strengthen the formulation, but it is generally assumed that they tend to produce more fractional LP values. Which relaxation

works best in practice shall be examined in the computational experiments in Section 5.

If RENS is used as a primal heuristic embedded in a complete solver, further modifications are necessary to obtain a good overall performance. When primal heuristics are considered as standalone solving procedures, e.g., the Feasibility Pump [3, 9, 21, 25, 26, 29, 32], the algorithmic design typically aims at finding feasible solutions for as many instances as possible, even if this takes substantial running time. However, if they are used as supplementary procedures inside a complete solver, the overall performance of the solver is the main objective. To this end, it is often worth sacrificing success on a small number of instances for a significant saving in average running time. The Stage 3 of the Feasibility Pump[1] is a typical example of a component that is crucial for its impressive success rate as a standalone algorithm, but it will not be applied when the Feasibility Pump is used inside a complete solver, see [10]. RENS principally is an expensive algorithm that solves an $\mathcal{NP}$-hard problem; therefore, the decision of when to call it should made carefully to avoid slowing down the overall solving process. The remainder of this section describes some algorithmic enhancements, most of which are concerned with identifying which subproblems are the most promising for calling RENS and on which subproblems it should be skipped.

First, RENS should only be called if the resulting sub-MINLP seems to be substantially easier than the original one. This means that at least a specific ratio of all integer variables, say $r_1 \in (0, 1)$, or a specific ratio of all variables including the continuous ones, say $r_2 \in (0, 1)$, should be fixed. The first criterion limits the difficulty of the discrete part of the sub-MINLP itself, the second one limits the total size of the relaxations that will have to be solved. For example, think of a MIP which consists of 20 integer and 10 000 continuous variables. Even if one fixes 50% of the integer variables, RENS would be a time-consuming heuristic since solving the LPs of the sub-MIP would be nearly as expensive as solving the ones of the original MIP. Since by propagation, fixing integer variables might also lead to fixed continuous variables, e.g., for variable bound constraints, we check the latter criterion only after presolving the subproblem.

Second, the sub-MINLP does not have to be solved to proven optimality. Therefore, we decided to use limits on the solving nodes and the so-called *stalling nodes* of the sub-MINLP. The absolute solving node limit l_1 is a hard limit on the maximum number of branch-and-bound nodes that should be processed. The stalling node limit l_2 indicates how many nodes should at most be processed without an improvement to the incumbent solution of the sub-MINLP.

Third, the partial solution of the sub-MINLP aims at finding a good primal solution quickly. Hence, algorithmic components that mainly improve the dual bound, such as cutting plane separation, and that are computationally very expensive, such as strong branching, can be disabled or reduced to a minimum. Further on this list are conflict analysis, pairwise presolving of constraints, probing and other LNS heuristics. As branching and node selection strategies we use inference branching and best estimate search, see, e.g. [1].

RENS could be either used as a pure start heuristic, calling it exclusively at the root node, or frequently throughout the branch-and-bound search to find rounded solutions of local LP optima. In particular when the integrality of the root LP relaxation falls below the minimum fixing ratio r_1, it seems reasonable to employ RENS at deeper levels of the tree where the number of fractional variables tends to be smaller. For the case of repeated calls of RENS, we implemented a few strategies to determine the points at which RENS should be called.

How often RENS should be called mainly depends on two factors: how expensive is it for a particular instance and how successful has it been in previous calls for that instance? The first can be estimated by the sum n_{RENS} of branch-and-bound nodes RENS used in previous calls

[1] Stage 3 of the Feasibility Pump solves (a reformulation of) the original MIP with a new objective function. It minimizes the distance to an infeasible point gained from the pumping algorithm; more precisely to the one which was closest to the polyhedron associated to the LP relaxation. For details, see [29].

in comparison to n_{all}, the number of branch-and-bound nodes already searched in the master problem. The second can be measured by the success rate $s = \frac{n_{\text{sols}}+1}{n_{\text{calls}}+1}$, where n_{calls} denotes the number of times RENS has been called and n_{sols} denotes the number of times it contributed an improving solution, respectively. In our implementation, we computed the stalling nodes limit as

$$l_2 = 0.3n_{\text{all}} \cdot s - n_{\text{RENS}} + 500 - 100n_{\text{calls}}.$$

The last term represents the setup costs for the subproblem which accrue even if subproblem solving terminates quickly. The offset of 500 nodes ensures that the limit is reasonable for the first few calls of RENS. We only start RENS if l_2 is sufficiently large.

In an LP-based branch-and-bound search, consecutive nodes tend to have similar LP optima. This is due to the similarity of the solved problems as well as to the warm-starting technique of the simplex algorithm, which is typically used for this purpose. Since similar LP optima most likely lead to similar results for the quite expensive RENS heuristic, it should not be called in consecutive nodes, but the calls should rather be spread equally over the tree. Therefore, we use a call frequency f: RENS only gets called at every f-th depth of the tree.

4 Experimental setup

This section proposes three computational experiments that evaluate the potential of RENS to find optimal rounded solutions, compare RENS to other rounding heuristics, and demonstrate the impact of RENS inside a full-scale branch-and-bound solver. We conduct these experiments on three different test sets of MIPs, MIQCPs, and MINLPs in order to analyze RENS on different classes of mathematical programs. All test sets are compiled from publically available libraries.

Few existing softwares solve nonconvex MINLPs to global optimality, including BARON [50], COUENNE [8], and LINDOGLOBAL [59]. Others, such as BONMIN [20] and SBB [60], guarantee global optimality only for convex problems, but can be used as heuristic solvers for nonconvex problems. For a comprehensive survey of available MINLP solver software, see [24, 27]. Recently, the solver SCIP [2, 61] was extended to solve nonconvex MIQCPs [17] and MINLPs [51] to global optimality. SCIP is currently one of the fastest noncommercial solvers for MIP [40, 45], MIQCP [45] and MINLP [51].

For all computational experiments, we used SCIP 2.1.1.1 compiled with SOPLEX 1.6.0 [55, 62] as LP solver, IPOPT 3.10 [52, 58] as NLP solver, and CPPAD 20110101 [57] as expression interpreter for evaluating general nonlinear constraints. The results were obtained on a cluster of 64bit Intel Xeon X5672 CPUs at 3.20GHz with 12 MB cache and 48 GB main memory, running an openSuse 11.4 with a GCC 4.5.1 compiler. Hyperthreading and Turboboost were disabled. In all experiments, we ran only one job per node to avoid random noise in the measured running time that might be caused by cache-misses if multiple processes share common resources.

Test sets. We used all instances from MIPLIB3.0 [18], MIPLIB2003 [4], and MIPLIB2010 [40] as MIP test set. We excluded instances `air03`, `ex9`, `gen`, `manna81`, `p0033`, `vpm1`, for which the optimum of the LP relaxation (after SCIP presolving) is already integral, instance `stp3d`, for which SOPLEX cannot solve the LP to optimality within the given time limit and instances `sp97ar`, `mine-166-5`, for which SOPLEX 1.6.0 fails in computing an optimal LP solution. This leaves 159 instances. We will refer to this test set as MIPLIB.

For MIQCP, we used the test set described in [15] that is comprised of instances from several sources. We excluded instances `ex1263`, `ex1265`, `sep1`, `uflquad-30-100`, for which the LP optimum is already integral (but in none of the cases feasible for the quadratic constraints), instances `nuclear14`, `isqp1`, `nuclearva`, for which the LP relaxation is unbounded, instance

200bar, for which SoPlex produces an error, 108bar, isqp0, for which scip's separation loop has not terminated within the time limit, and those 18 instances that are linear after scip presolving, see [15]. This test set contains 70 instances.

We further tested rens on general MINLPs from MINLPLib [23], excluding those that are MIQCPs, that are linear after scip presolving, or that contain expressions which cannot be handled by scip, e.g., sin and cos. We also excluded 4stufen, csched1a, st_e35, st_e36, waters, for which the optimum of the LP relaxation is integral, and instances csched2, minlphix, uselinear, for which the LP relaxation is unbounded, leaving 105 instances. It remains to be said that this test set is not as balanced as the others, since there are many instances of similar type.[2]

Analyzing roundability and computing optimal roundings. In a first test, we employ rens to analyze the roundability of an optimal relaxation solution. For this, we run rens without any node limits or variable fixing thresholds on the test sets described above. A time limit of two hours, however, was set for solving the rens subproblem.

We used the optimum of the LP relaxation as starting point for the MIP test. We compare the performance of rens using the "original" LP optimum before the cutting plane separation loop versus the one after cuts. One question of interest here is how the integrality of the LP solution interacts with the feasibility of the sub-MIP. The desired situation is that the LP solution contains a lot of integral values, but still gives rise to a feasible rens problem.

For the MIQCP and the MINLP test run, we further evaluate how different types of relaxations, the LP and the NLP relaxation, behave w.r.t. the roundability of their optima and the quality of the rounded solutions. The results shall give an insight into which solutions should be used as starting points for rens and other primal heuristics. Here, the performance in terms of running time of the rens heuristic has to be weighed up against the success rate and quality of solutions produced with different relaxations.

Evaluating the performance of rounding heuristics. In a second test, we use rens for the analysis of several rounding heuristics. The results shall give an insight into how often these heuristics find a feasible rounding and how good the quality of this solution is w.r.t. the optimal rounding.

All considered rounding heuristics iteratively round all variables that take a fractional value in the optimum of the relaxation. One rounding is performed per iteration step, without resolving the relaxation.

- *Simple Rounding* only performs roundings, that maintain feasibility;
- *ZI Round* conducts roundings, using row slacks to maintain primal feasibility;
- *Rounding* conducts roundings, that potentially violate some constraints and reduces existing violations by further roundings;

ZI Round and Rounding both are extensions of Simple Rounding. Both are more powerful, but also more expensive in terms of running time.

For more details on ZI Round, see [53], for details on the other rounding heuristics implemented in scip, see [10].

Note that these heuristics are quite defensive, in the sense that they often round opposite to the variable's objective function coefficient and sacrifice optimality for feasibility. Hence, we do not expect them to often detect the optimal rounding computed by rens. The question is rather

[2]This holds, to a certain extent, for all general MINLP test sets that the author is aware of.

for how many of the roundable instances these heuristics find any feasible solution and only as a second point how big the gap to the optimal rounding is.

rens compared to other primal heuristics. In a third test, we compare RENS to other primal heuristics embedded in SCIP and called at the root node. We measure RENS against the complete portfolio of root node heuristics and against the single best heuristic (as implemented in SCIP). In the case of MIP, this was the Feasibility Pump [29, 3]; in the case of MIQCP and MINLP, this turned out to be Undercover [13, 14].

SCIP applies eleven primal heuristics at the root node: three rounding heuristics (see previous experiment), three propagation heuristics, a trivial one, a feasibility pump, an improvement heuristic, a large neighborhood search, and a repair heuristic. The latter two only come into play for nonlinear problems. This experiment is done to check whether SCIP is competitive with heuristics that are more involved than the rounding heuristics from the previous experiment.

Impact of rens on the overall performance of scip. In our final experiment, we evaluate the usefulness of RENS when applied as a primal heuristic inside a branch-and-bound solver. For comparison see the RINS algorithm [28], an improvement heuristic which is applied in CPLEX and GUROBI. The advantage of RENS in contrast to RINS is that it does not require a given primal solution and that it always fixes at least the same number of variables as RINS, if applied to the same relaxation solution. The advantage of RINS is that the RINS subproblem is guaranteed to contain at least one feasible solution, namely the given starting solution.

To assess RENS as a primal heuristic, we run SCIP with RENS applied exclusively as a root node heuristic and SCIP with RENS applied both at the root and throughout the search. For this experiment, we used a reduced version of RENS which requires a minimal percentage of variables to be fixed and which stops after a certain number of branch-and-bound nodes, see Section 3. For comparison, we ran SCIP with RENS deactivated.

The main criteria to analyze in this test are the impact of RENS on the quality of the primal bound early in the search and the impact of RENS on the overall performance. While we hope for improvements in the former, a major improvement in the latter is not to be expected. Different studies show that the impact of primal heuristics on time to optimality often is slim. Bixby et al. report a deterioration of only 9% if deactivating all primal heuristics in CPLEX 6.5, Achterberg [1] presents a performance loss of 14% when performing a similar experiment with SCIP 0.90i, in [10] differences of at most 5% for deactivating single primal heuristics are given. Therefore, a good result for this experiment would be an improvement on the primal bound side, coming with no deterioration to the overall performance.

5 Computational results

As a first test, we ran RENS without node or variable fixing limits, to evaluate its potential to find optimal roundings of optimal LP and NLP solutions.

The results for MIP can be seen in Tables 4 and 5, those for MIQCP in Tables 6 and 7, those for MINLP in Tables 8 and 9; aggregated results can be found in Table 1. Each table presents the names of the instances, Int, the percentage of integer variables that were fixed by RENS, All, the percentage of all variables that were fixed after presolving of the RENS subproblem, TimeS, the time SCIP needed before RENS was called, Time and Nodes, the running time and the number of branch-and-bound nodes needed to solve the subproblem to optimality, Solution, the best solution found in the RENS subproblem, and Found At, the node in the subproblem's branch-and-bound

tree at which it has been found. Note that these values are rounded, e.g., the 100.0% given in column Int of Table 4 for nw04 represents a ratio of $^{87460}/_{87482}$.

If the subproblem was proven to be infeasible or no solution was found within the time limit, this is depicted by an "–" in the column Solution. When the time limit of two hours was hit in the RENS subproblem, this is indicated by the term limit in the Time column. Hence, for all instances that do not hit the time limit, the column Solution depicts the proven optimal rounding of the relaxation solution and "–" indicates that it was proven that no feasible rounding exists. Instances for which the optimal rounding is an optimal solution of the original MINLP are marked by a star.

The correlation between the percentage of fixed variables and the success of RENS is depicted in Figures 3–6. Each instance is represented by a cross, with the fixing rate being the x-coordinate, and 0 or 1 representing success or failure as y-coordinate. The dotted blue line shows a moving average taken over ten consecutive points and the dashed red line shows a moving average taken over 30 consecutive points. A thin gray line is placed at the average success rate taken over all instances of the corresponding test set.

If we have to average running times or number of branch-and-bound nodes, we use a *shifted geometric mean*. The shifted geometric mean of values $t_1, \ldots, t_n$ with shift s is defined as $\sqrt[n]{\prod(t_i + s)} - s$. We use a shift of $s = 10$ for time and $s = 100$ for nodes in order to reduce the effect of very easy instances in the mean values. Further, using a *geometric* mean prevents hard instances at or close to the time limit from having a huge impact on the measures. Thus, the shifted geometric mean has the advantage that it reduces the influence of outliers in both directions.[3] In the given mean numbers, instances hitting the time limit are accounted for with the time limit and the number of processed nodes at termination.

In Table 1, Columns "> 90%" and "avg" show the number of instances for which more than 90% of the integer variables were integral and the average percentage of integer variables taking integral values, respectively. Column "succ" depicts the percentage of instances for which RENS found a feasible rounding. Columns "nodes" and "time (s)" give the shifted geometric means of the branch-and-bound nodes and running time required for solving the RENS subproblem.

Unless otherwise noted, the term variables always refers to *integer* variables for the remainder of this section.

Computing optimal roundings: MIP. In Table 4, we see that for roughly one third ($^{55}/_{159}$) of the instances, more than 90% of the variables took an integral solution in the optimal LP solution. In contrast to that, there are only 22 instances for which the portion of integral solution values is less than 40%. The average percentage of variables with integral LP solution value is 71.7%. There are a few cases with many continuous variables for which fixing the majority of the integer variables did not result in a large ratio of all variables being fixed, see, e.g., dsbmip or p5_34. This is the reason that we will use two threshold values for later tests, see Section 3.

For 59.7% ($^{95}/_{159}$) of the instances, RENS found a feasible rounding of the LP optimum. For 15 of these instances, the RENS subproblem hit the time limit, eleven of them are from MIPLIB 2010. For the remaining 80 instances, the solutions reported in Table 4 are the optimal roundings of the given starting solutions. For 34 instances, the optimal rounding coincides with the global optimal solution.

We further observe that the success rate is only weakly correlated to the ratio of fixed variables. The success rate on the instances with more than 90% fixed variables was nearly the same as on the whole test set, namely 58.2%. This is an encouraging result for using RENS as a start heuristic inside a complete solver: very small subproblems contain feasible solutions.

[3] For a detailed discussion of the shifted geometric mean, see Achterberg [1, Appendix A3].

Figure 3: Moving averages of success rate, MIPLIB instances, after cuts

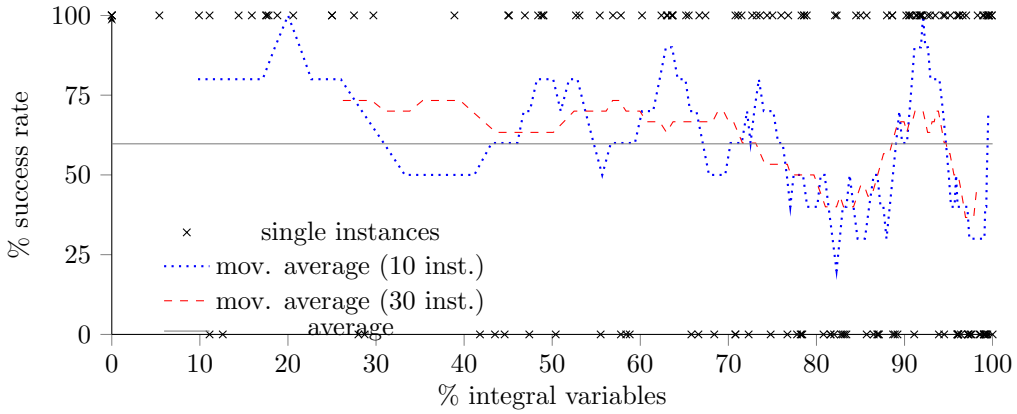
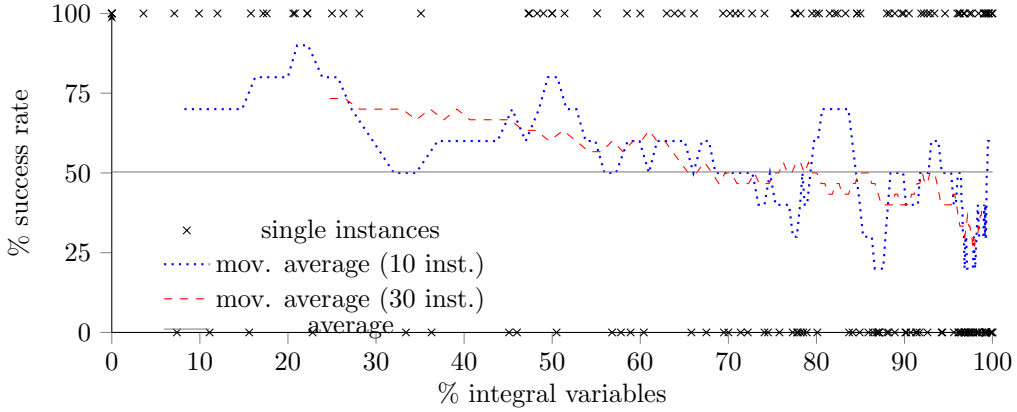


Figure 4: Moving averages of success rate, MIPLIB instances, before cuts

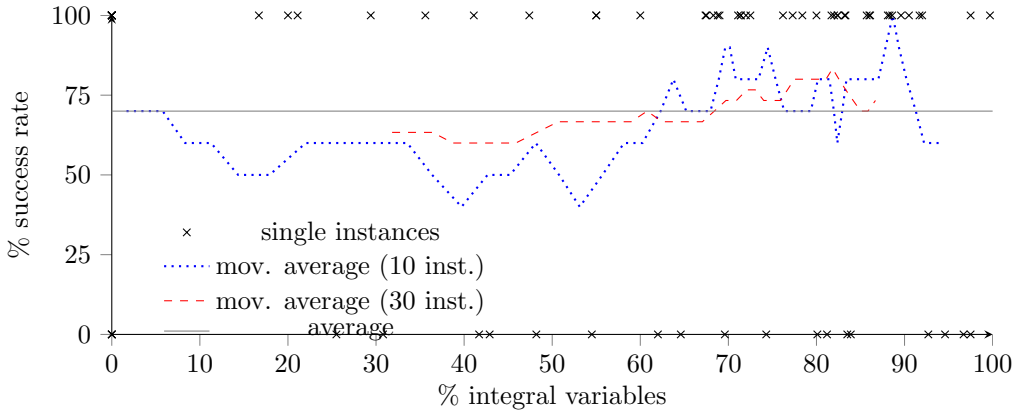


The connection between the fixing rate and the success rate is also depicted in Figure 3. We see that the success rate decreases slightly, at about 75% fixed variables, but the difference between low and high fixing rates is not huge.

We further observe that proving the non-existence of a feasible rounding is relatively easy in most cases. For 59 out of 64 infeasible rounding subproblems, infeasibility could be proven in presolving or while root node processing of the subproblem. There is only one instance, `pigeon-10`, for which proving infeasibility takes more than 600 nodes. Considering the running time, infeasibility could be proven in less than a second in 56 of 64 cases, with only one instance, `app1-2`, taking more than 15 seconds. The instance `neos-1601936` is the only one for which feasibility could not be decided within the given time limit; hence, it is the only instance for which we could not decide whether the optimal LP solution is roundable or not.

The results for using the LP optimum before cutting plane separation are shown in Table 5. Even more instances, 62 compared to 55, have an integral LP solution for more than 90% of the variables. However, there is one more (24 vs. 23) instance, for which the portion of integral solution values is less than 40%. Contrary to what one might expect, the average percentage of variables with integral LP value is hardly affected by cutting plane separation: it is 73.6% before

Figure 5: Moving averages of success rate, MIQCP instances, LP sol., after cuts



separation and 71.7% after.

The number of instances for which RENS found a solution, however, goes down: 80 instead of 95, which is only half of the test set. This is particularly due to those instances with many variables that take an integral value. Consequently, the success rate of RENS drops with an increase in the ratio of fixed variables. When RENS is called before cutting planes are added, fewer of the optimal roundings are optimal solutions to the original problem: 20 compared to 34, when called after cuts.

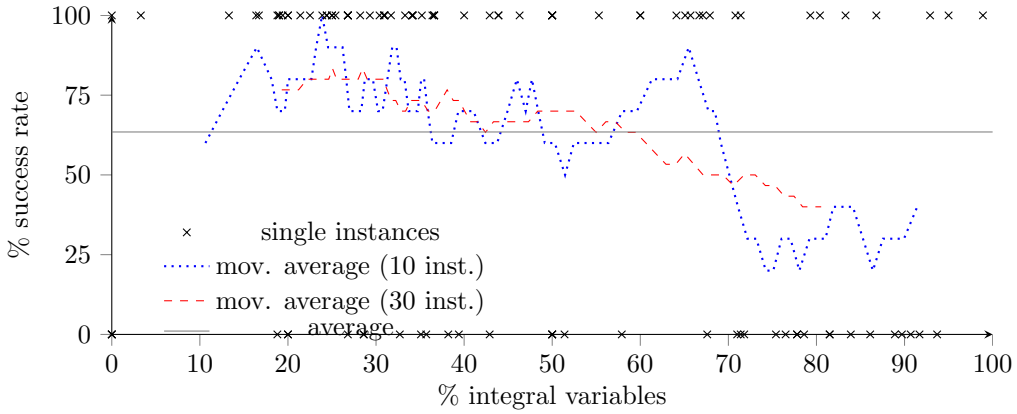
We conclude that, although the fractionality is about the same, LP solutions before cuts are less likely to be roundable and the rounded solutions are often of inferior quality. In other words: before cutting planes, integral solution values are more likely to be misleading (in the sense that they cannot be extended to a good feasible solution). This is an important result for the design of primal heuristics in general and confirms the observation that primal heuristics work better after cutting plane separation, see, e.g., [32].

Computing optimal roundings: MIQCP. For MIQCP, we tested RENS with LP solutions, see Table 6, and with NLP solutions, see Table 7, as starting points. We also experimented with the LP solution before cuts; the results were much worse and are therefore not shown.

The ratio of integral LP values is smaller compared to the MIP problems: there are only 9 out of 70 instances for which more than 90% of the variables were integral, but there are 10 instances for which all variables were fractional. Note that this does not necessarily mean that the RENS sub-MIQCP is identical to the original MIQCP, cf. the presence of general integer variables. In this case, the RENS subproblem corresponds to the original problem intersected with the integral lattice-free hypercube around the starting solution. On average, 59.9% of the variables took an integral value. The success rate of RENS is even better than for MIPs: In 49 out of 70 instances (70%), RENS found a feasible rounding. Note that this is not due to the 10 instances for which all variables were fractional: three of them also fail. Moreover, the success rate appears not to depend on the percentage of fixed variables, see Figure 5.

Deciding feasibility, however, seems to be more difficult. Out of ten instances hitting the time limit, there were eight for which RENS did not find a feasible rounding. For 13 instances, infeasibility of the rounding problem was proven, mostly in presolving or within a few branch-and-bound nodes. Nine times, the optimal rounding was identical to the optimal solution of the MIQCP.

Figure 6: Moving averages of success rate, MINLP instances, LP sol., after cuts



The next observation we made is that the NLP solution tends to be much less integral than the LP solution, on average only 13.8% of the variables take an integral value, see Table 7 and Figure 5. This is due to the fact that in our experiments the LP solution was computed with the simplex algorithm which tends to leave variables at their bounds, whereas the NLP solution was computed with an interior point algorithm that tends to choose values from the interior of the variables' domains.

Surprisingly, this did not enhance the roundability. For 48 instances, RENS found a feasible rounding of the NLP optimum, compared to 49 for the LP. Worth mentioning, this was nearly the same set of instances, and there were 46 on which both versions found a solution. The solution quality, however, was typically better when using an NLP solution: 27 times the NLP solution yielded a better rounding, only once the LP was superior. 26 times, the optimal rounding was even an optimal solution of the original MIQCP.

The higher fractionality of the NLP relaxation is expressed in a much larger search space. In shifted geometric mean, RENS processed 628 search nodes if starting from an LP solution, 7078 if starting from an NLP solution. The geometric mean of the running time (Time) is roughly 5.5 times larger: 30.9 vs. 168.1 seconds.

We conclude that the same observation holds as in the MIP case: small subproblems generate high-quality feasible solutions. Although the solution quality is improved by using an NLP relaxation, the computational overhead and the success rate are not encouraging to make this a standard setting if using RENS inside a complete solver.

Computing optimal roundings: MINLP. For MINLP, we again compared two versions of RENS: one using the LP solution and one using the NLP solution as starting point, see Tables 8 and 9, respectively. For the same reason as in the MIQCP case we omitted the results for the LP solution before cuts.

The integrality of the LP solutions is comparable to the MIQCP case. On average, 63.5% of the variables take an integral value; there are 6 out of 105 instances for which more than 90% of the variables are integral, and only four instances for which all variables are fractional. For this test set, we see a clearer connection between the ratio of fractional variables and the success rate of RENS. The more variables are integral, the lower the chance for RENS to succeed, see Figure 6.

For seven instances, the RENS subproblem hit the time limit of two hours, always without having found a feasible solution. Overall, 65 out of 105 (62%) of the LP solutions proved to be

Table 1: Computing optimal roundings (aggregated results)

	integrality			comp. effort	
	>90%	avg	succ	nodes	time (s)
MIP + cuts	55/159	71.7%	59.7%	814.4	22.6
MIP − cuts	62/159	73.6%	50.3%	719.9	21.7
MIQCP (LP)	9/70	59.9%	70.0%	627.7	30.9
MIQCP (NLP)	1/70	13.8%	68.6%	7078.8	168.1
MINLP (LP)	6/105	63.5%	61.9%	11175.6	83.0
MINLP (NLP)	1/105	15.0%	69.5%	93908.0	262.7

roundable, which is similar to the MIP results. In all cases, RENS found the optimal rounding. Generally, RENS needs much more nodes to solve the rounding problem as compared to the other tests.

Using the NLP instead of the LP relaxation slightly increases the success rate: 73 times, RENS finds a feasible rounding. As for MIQCPs, the quality is typically better (37 vs. 2 times), which comes with a much lower integrality of 15% on average, 68 instances having all variables fractional, and a huge increase in running time: a factor of more than three in shifted geometric mean.

Computing optimal roundings: summary. Interestingly, the fractionality and roundability of LP solutions is very similar for MIPs, MIQCPs and MINLPs: on average, only 30–40% of the variables are fractional and for 60–70% of the instances RENS found a feasible rounding. We further observed that most often the RENS subproblem could be solved to proven optimality and that the success rate of RENS is only weakly correlated to the fractionality. These three insights are very encouraging for applying RENS as a start heuristic inside a complete solver, see below. A summary of the results on computing optimal roundings can be found in Table 1.

We further performed a McNemar test to analyze the statistical significance of the results. As null hypothesis we assume that the LP and the NLP solution (or the LP before and after cuts) are equally likely to yield a feasible rounding. For the MIP test set, the null hypothesis gets rejected with a p-value of 0.0011 and for MINLP with 0.0114. For MIQCP, the p-value is 0.6547. This means that for MIP the LP solution after cuts is more likely to be roundable with very high probability, for MINLP the NLP solution is more promising with high probability, for MIQCP there is no statistically significant difference.

We conclude that the solutions found by RENS are usually better when it is applied after cutting plane separation and that using an NLP instead of an LP relaxation does not give a good trade-off between solution quality and running time: it might be better, but the computational overhead is huge.

Analyzing rounding heuristics. Our next experiment compares RENS applied to the LP solution after cuts with the three pure rounding heuristics that are implemented in SCIP. The results for the MIPLIB instances are shown in Table 10. Instances for which none of the compared methods could provide a solution are omitted in the presentation.

As implied by definition, the solutions found by RENS (if the subproblem has been solved to optimality) are always better or equal to the solutions produced by any rounding heuristic. As expected, the solution quality of Rounding and ZI Round is always better or equal to Simple Rounding, and ZI Round often is superior to Rounding. Since Simple Rounding,

Rounding, and ZI Round all endeavor to feasibility and neglect optimality, it is not too surprising that there are only three instances, for which Simple Rounding and Rounding find an optimal rounding; four in the case of ZI Round.

A comparison of the number of solutions, however, shows that there is a big discrepancy between the number of instances which have a roundable LP optimum (95) and the number of instances for which these heuristics succeed (37 for ZI Round, 36 for Rounding, and 27 for Simple Rounding). Of course, this has to be seen under the fact that these heuristics are much faster than RENS. The maximum running time was attained by Rounding on instance `opm2-z7-s2`; it was only 0.09 seconds.

For the MIQCP and MINLP test sets, the situation was even more extreme. The rounding heuristics were unable to produce a feasible solution for any of the instances – even though the previous experiments proved that 60–70% of the LP solutions are roundable. This is most likely due to the special design of these heuristics – they solely work on the LP relaxation – and demonstrates the need for rounding heuristics that take the special requirements of nonlinear constraints into consideration.

rens compared to other primal heuristics. This experiment compares RENS to other primal heuristics embedded in SCIP and called at the root node. For each of the three test sets, we evaluated three different settings of SCIP: One for which all default root node heuristics except RENS are employed, one for which only RENS is called, and one for which only the Feasibility Pump (for MIP) or only Undercover (for MIQCP and MINLP) is used.

Based on the results from our first experiment, considering the running times and the node numbers at which the RENS subproblems find their optimal solutions, we decided to use 50% as a threshold value for r_1, the minimal fixing rate for integer variables, in this run. The minimal fixing rate for all variables r_2 was set to 25%. We used an absolute node limit l_1 of 5000 and computed the stalling node limit l_2 as given in Section 3. Because of the long running times, we refrained from using an NLP relaxation, although it might produce better solutions. We always used the LP solution after cutting planes as a starting solution.

The results are shown in Tables 11–13. Instances for which none of the compared methods could provide a solution are omitted in the presentation.

We observe that for all three test sets, RENS alone is inferior to the portfolio of root node heuristics, but superior to the single best heuristic in terms of problem instances for which a solution could be found. For the MIP instances, SCIP's root node heuristics found feasible solutions for 106 instances, RENS (with the described settings) for 56, the Feasibility Pump for 51. For MIQCP, the portfolio succeeded 56 times, RENS 33 times, Undercover 29 times. For MINLP the result was 28 for all, 12 for RENS, 6 for Undercover. Note that on this test set, as is typical for nonconvex MINLPs, finding a feasible solution is generally harder than for MIPs. Other solvers perform comparably: we additionally performed this root node test with the default settings and a time limit of two hours for COUENNE 0.4 [8] and BARON 11.1 [50]. They found feasible solutions for 35 and 40 instances, respectively.

There were two MIP instances, two MIQCP instances, and three MINLP instances, for which RENS found a feasible solution but the other SCIP root node heuristics did not. For a further 40, 17, and 5 instances, respectively, the solution found by RENS was better than the best solution produced by the other heuristics. We conclude that RENS is a valuable extension of SCIP's primal heuristic portfolio. Further, in terms of the number solutions it produced when run as the only heuristic, it is comparable to other state-of-the-art primal heuristics, such as the Feasibility Pump (in an embedded version, compare Section 3) or Undercover.

Table 2: RENS as primal heuristic inside SCIP (aggregated results), numbers of instances for which RENS was called and succeeded at least once

	at root		in tree	
	called	found	called	found
MIP (of 160)	124	63	154	87
MIQCP (of 70)	45	31	60	42
MINLP (of 105)	45	9	99	39

Table 3: RENS as primal heuristic inside SCIP (aggregated results), computational effort

	arithmetic		geometric		shifted geom	
	nodes	time(s)	nodes	time(s)	nodes	time(s)
MIP No RENS	1 446 078	2461.4	7 155	220.3	11 248	377.2
MIP Root RENS	1 442 400	2427.0	5 870	209.6	10 390	366.3
MIP Tree RENS	1 443 404	2414.3	5 810	209.4	10 346	365.8
MIQCP No RENS	659 740	2872.3	3 823	84.5	6 457	229.9
MIQCP Root RENS	677 123	2927.0	3 742	86.4	6 361	232.0
MIQCP Tree RENS	664 117	2888.6	3 561	86.2	6 193	229.9
MINLP No RENS	2 338 903	3274.5	45 334	288.0	58 758	466.5
MINLP Root RENS	2 324 208	3274.7	44 723	291.4	58 406	467.1
MINLP Tree RENS	1 925 902	3168.7	38 568	267.9	51 066	431.3

Impact of rens on the overall performance of scip. Finally, we evaluate whether a reduced version of the full RENS algorithm is suited to serve as a primal heuristic applied inside a complete solver. We use the same threshold settings as in the previous experiment. For this experiment, interactions of different primal heuristics among each other and with other solver components come into play. SCIP applies eleven primal heuristics at the root node. Of course, a primal heuristic called prior to RENS might already have found a solution which is better than the optimal rounding, or in an extreme case, the solution process might already terminate before RENS is called. Further, any solution found before RENS is called might change the solution path. It might trigger variable fixings by dual reductions, which lead to a different LP and hence to a different initial situation for RENS.

The results are shown in Tables 14–16. We compare SCIP without the RENS heuristic (No RENS) against SCIP with RENS applied at most once at the root node (Root RENS) and SCIP with RENS applied at every tenth depth of the branch-and-bound tree (Tree RENS). Columns Nodes and Time show the number of branch-and-bound nodes and the running time SCIP needs to solve an instance to proven optimality. If a limit was hit, this is indicated by the term limit in the time column and the node number at which the solution process stopped is preceded by a '>'-symbol. At the bottom of the table, the arithmetic means, the geometric means, and the shifted geometric means of the number of branch-and-bound nodes and the running time are given.

A summary of the results is given in Tables 2 and 3. The Columns "called" and "found" in Table 2 show for how many instances RENS was called and found a feasible solution, respectively. Table 3 depicts the arithmetic means, the geometric means, and the shifted geometric means of the number of branch-and-bound nodes and the running time for each combination of the three different settings and the three test sets.

First, let us consider the results for MIP, see Table 14. Due to the a-priori limits, RENS was called at the root node for only 124 out of the 160 instances. Out of these, RENS found a feasible solution in 63 cases, which corresponds to a success rate of 50%, compared to 59% without any limits, see above. In 61 cases, this solution was the best solution found at the root node. Considering that there are ten other primal heuristics applied at the root node, this appears to be a very strong result. When RENS was additionally used during search, it was called on 154 instances, finding feasible solutions for 87 of them.

As is typical for primal heuristics, the impact on the overall performance is not huge. Nevertheless, we see that both versions, calling RENS only at the root and all over the tree, give slight decreases in the arithmetic and geometric means of the node numbers and the running time. Both versions were about 3% faster and took 8% less nodes in shifted geometric mean. For the time-outed instances, Root RENS and Tree RENS provided a better primal bound than No RENS eight and nine times, respectively, whereas both were inferior in two cases.

For the MIQCP test set, RENS was called at the root for 45 out of 70 instances, finding a feasible solution in 31 cases. This was always the best solution SCIP found at the root node. The overall performance was about the same: the running time stayed constant for Tree RENS and was increased by less than one percent for Root RENS, whereas the number of branch-and-bound nodes was reduced by 7% and 2%, respectively. When RENS is called during search tree processing, there are four instances with a better primal bound at timeout, once it was worse. For calling RENS exclusively at the root, this ratio was 2:0. Also, there is one instance, namely `nuclear14a`, for which only Tree RENS provided a feasible solution.

For MINLP, the lower success rate for the root LPs with large ratios of integral variables is confirmed by this experiment. For 45 out of 105 instances, RENS was called, but in only 9 cases it could improve the incumbent solution. Interestingly, the version that calls RENS during the tree performs really well. There were 42 instances, for which RENS could improve the incumbent at least once during search, `ghg_3veh` being the front-runner with 27 improving solutions in 44 calls of RENS.

The overall performance reflects that situation. The Root RENS setting shows the same behavior as No RENS, the running time is nearly equal on average and in geometric mean, the number of branch-and-bound nodes goes down by one percent, there are hardly any instances for which we see any change in performance. For Tree RENS, however, the geometric mean of the running time and the number of branch-and-bound nodes goes down by 8% and 13%, respectively. One might argue that this is mainly because of `enpro48pb` and `fo8_ar4_1` which show a dramatic improvement in performance. But even if we excluded these two instances (and for fairness reasons also `enpro48` and `enpro56pb`, two outliers in the opposite direction), the mean performance gain is 3% for running time and 8% for number of branch-and-bound nodes.

We further performed a variant of the Wilcoxon signed rank test to analyze the statistical significance of the results, using the STATS package of the SCIPY project [39]. We ranked the results by the running time factors per instance and calculated one rank sum from the improving instances and one from those which showed a degradation. Instances that showed no or hardly any performance difference (less than one second or less than 1%) were excluded. As null hypothesis, we assume that a version of SCIP using RENS at the root or throughout the tree does perform equally w.r.t. running time as SCIP without RENS. For the MIP test set, the null hypothesis gets rejected with a p-value of 0.0236 (for Root RENS) and 0.0178 (for Tree RENS) which is below the standard threshold of 0.05 used as level of significance. Not surprisingly, the results for MIQCP indicate that there are no performance differences for this test set: the p-values are 0.6465 and 0.8753 for Root RENS and for Tree RENS, respectively. For MINLP, p-values of 0.3980 and 0.2862 are achieved. Although failing to reject the null hypothesis when a standard threshold is applied, at least the latter could be taken as an indicator that it is more likely that the results are not

simply acquired by chance.

Altogether, these experiments show that RENS, in particular for MIP and MIQCP, helps to improve the primal bound at the root node, and hence the initial gap before the branch-and-bound search starts. Applying RENS exclusively at the root node had a neutral to slightly positive effect on the overall performance, while giving a user the advantage of finding good solutions early. Applying RENS throughout the search was at least as good for all three test sets and showed a nice improvement in the case of MINLP– which was partly due to two outliers. Consequently, RENS is used in the default settings of SCIP. Furthermore, versions of RENS have been recently integrated into BONMIN [20] and CBC [56].

6 Conclusion

We introduced RENS, a large neighborhood search algorithm that, given a MIP or an MINLP, solves a subproblem whose solution space is the feasible roundings of a relaxation solution. We showed that most MIP, MIQCP, and MINLP instances have roundable LP and NLP optima and in most cases, the optimal roundings can be computed efficiently. Surprisingly, the roundability seems not to be related to the fractionality of the starting solution. Knowing the optimal roundings provides us with a benchmark for rounding heuristics; we discovered that the rounding heuristics implemented in SCIP often fail in finding a feasible solution, even though the provided starting point is roundable. They rarely find the optimal rounding.

We further investigated the impact of a reduced version of RENS if applied as a primal heuristic inside a complete solver. RENS directly helps to improve the primal bound known at the root node. The impact on the overall performance is minor but measurable, which is typical for primal heuristics.

RENS is part of the SCIP standard distribution and employed by default. The implementation presented in this article can be accessed in source code at [61].

Acknowledgments

Many thanks go to Ambros M. Gleixner and Daniel E. Steffy for their thorough proof-reading and to two anonymous reviewers for their helpful comments. This research has been supported by the DFG Research Center MATHEON *Mathematics for key technologies*[4] in Berlin.

References

[1] T. Achterberg. *Constraint Integer Programming.* PhD thesis, Technische Universität Berlin, 2007.

[2] T. Achterberg. SCIP: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[3] T. Achterberg and T. Berthold. Improving the feasibility pump. *Discrete Optimization*, Special Issue 4(1):77–86, 2007.

[4] T. Achterberg, T. Koch, and A. Martin. MIPLIB 2003. *Operations Research Letters*, 34(4):1–12, 2006. `http://miplib.zib.de`.

[4] `http://www.matheon.de`

[5] E. Balas. Facets of the knapsack polytope. *Mathematical Programming*, 8:146–164, 1975.

[6] E. Balas, S. Ceria, M. Dawande, F. Margot, and G. Pataki. Octane: A new heuristic for pure 0-1 programs. *Operations Research*, 49, 2001.

[7] E. Balas, S. Schmieta, and C. Wallace. Pivot and shift - a mixed integer programming heuristic. *Discrete Optimization*, 1(1):3–12, June 2004.

[8] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods & Software*, 24:597–634, 2009.

[9] L. Bertacco, M. Fischetti, and A. Lodi. A feasibility pump heuristic for general mixed-integer problems. *Discrete Optimization*, Special Issue 4(1):77–86, 2007.

[10] T. Berthold. Primal heuristics for mixed integer programs. Diploma thesis, Technische Universität Berlin, 2006.

[11] T. Berthold. Heuristics of the branch-cut-and-price-framework SCIP. In J. Kalcsics and S. Nickel, editors, *Operations Research Proceedings 2007*, pages 31–36. Springer-Verlag, 2008.

[12] T. Berthold, T. Feydy, and P. J. Stuckey. Rapid learning for binary programs. In A. Lodi, M. Milano, and P. Toth, editors, *Proc. of CPAIOR 2010*, volume 6140 of *LNCS*, pages 51–55. Springer, June 2010.

[13] T. Berthold and A. M. Gleixner. Undercover – a primal heuristic for MINLP based on sub-MIPs generated by set covering. In P. Bonami, L. Liberti, A. J. Miller, and A. Sartenaer, editors, *Proceedings of the EWMINLP*, pages 103–112, April 2010.

[14] T. Berthold and A. M. Gleixner. Undercover – a primal MINLP heuristic exploring a largest sub-MIP. ZIB-Report 12-07, Zuse Institute Berlin, 2012. Accepted for publication in Mathematical Programming.

[15] T. Berthold, A. M. Gleixner, S. Heinz, and S. Vigerske. Analyzing the computational impact of MIQCP solver components. *Numerical Algebra, Control and Optimization*, 2(4):739–748, 2012.

[16] T. Berthold, S. Heinz, M. E. Pfetsch, and S. Vigerske. Large neighborhood search beyond MIP. In L. D. Gaspero, A. Schaerf, and T. Stützle, editors, *Proceedings of the 9th Metaheuristics International Conference (MIC 2011)*, pages 51–60, 2011.

[17] T. Berthold, S. Heinz, and S. Vigerske. Extending a CIP framework to solve MIQCPs. In J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 427–444. Springer, 2011.

[18] R. E. Bixby, S. Ceria, C. M. McZeal, and M. W. Savelsbergh. An updated mixed integer programming library: MIPLIB 3.0. *Optima*, (58):12–15, 1998.

[19] R. E. Bixby, M. Fenelon, Z. Gu, E. Rothberg, and R. Wunderling. MIP: Theory and practice – closing the gap. In M. Powell and S. Scholtes, editors, *Systems Modelling and Optimization: Methods, Theory, and Applications*, pages 19–49. Kluwer Academic Publisher, 2000.

[20] P. Bonami, L. Biegler, A. Conn, G. Cornuéjols, I. Grossmann, C. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Disc. Opt.*, 5:186–204, 2008.

[21] P. Bonami, G. Cornuéjols, A. Lodi, and F. Margot. A feasibility pump for mixed integer nonlinear programs. *Mathematical Programming*, 119(2):331–352, 2009.

[22] P. Bonami and J. Gonçalves. Heuristics for convex mixed integer nonlinear programs. *Computational Optimization and Applications*, pages 1–19, 2010.

[23] M. Bussieck, A. Drud, and A. Meeraus. MINLPLib – a collection of test models for mixed-integer nonlinear programming. *INFORMS Journal on Computing*, 15(1):114–119, 2003.

[24] M. R. Bussieck and S. Vigerske. MINLP solver software. In J. J. Cochran, L. A. Cox, P. Keskinocak, J. P. Kharoufeh, and J. C. Smith, editors, *Wiley Encyclopedia of Operations Research and Management Science*. Wiley and Sons, Inc., 2010.

[25] C. d'Ambrosio, A. Frangioni, L. Liberti, and A. Lodi. Experiments with a feasibility pump approach for nonconvex MINLPs. In P. Festa, editor, *Experimental Algorithms*, volume 6049 of *Lecture Notes in Computer Science*, pages 350–360. Springer Berlin / Heidelberg, 2010.

[26] C. d'Ambrosio, A. Frangioni, L. Liberti, and A. Lodi. A storm of feasibility pumps for nonconvex MINLP. *Mathematical Programming*, 136:375–402, 2012.

[27] C. d'Ambrosio and A. Lodi. Mixed integer nonlinear programming tools: a practical overview. *4OR: A Quarterly Journal of Operations Research*, 9:329–349, 2011.

[28] E. Danna, E. Rothberg, and C. L. Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming A*, 102(1):71–90, 2004.

[29] M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. *Mathematical Programming A*, 104(1):91–104, 2005.

[30] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming B*, 98(1-3):23–47, 2003.

[31] M. Fischetti and M. Monaci. Proximity search for 0-1 mixed-integer convex programming. Technical report, DEI, University of Padova, 2012.

[32] M. Fischetti and D. Salvagnin. Feasibility pump 2.0. *Mathematical Programming C*, 1:201–222, 2009.

[33] S. Ghosh. DINS, a MIP improvement heuristic. In M. Fischetti and D. P. Williamson, editors, *Integer Programming and Combinatorial Optimization (IPCO 2007)*, volume 4513 of *LNCS*, pages 310–323, 2007.

[34] F. Glover and M. Laguna. General purpose heuristics for integer programming – part I. *Journal of Heuristics*, 2(4):343–358, 1997.

[35] F. Glover and M. Laguna. General purpose heuristics for integer programming – part II. *Journal of Heuristics*, 3(2):161–179, 1997.

[36] F. Glover, A. Løkketangen, and D. L. Woodruff. Scatter search to generate diverse MIP solutions. *OR Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research*, 2000.

[37] P. L. Hammer, E. L. Johnson, and U. N. Peled. Facets of regular 0-1 polytopes. *Mathematical Programming*, 8:179–206, 1975.

[38] P. Hansen, N. Mladenović, and D. Urošević. Variable neighborhood search and local branching. *Computers and Operations Research*, 33(10):3034–3045, 2006.

[39] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[40] T. Koch, T. Achterberg, E. Andersen, O. Bastert, T. Berthold, R. E. Bixby, E. Danna, G. Gamrath, A. M. Gleixner, S. Heinz, A. Lodi, H. Mittelmann, T. Ralphs, D. Salvagnin, D. E. Steffy, and K. Wolter. MIPLIB 2010. *Mathematical Programming Computation*, 3(2):103–163, 2011.

[41] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.

[42] L. Liberti, N. Mladenović, and G. Nannicini. A recipe for finding good solutions to MINLPs. *Mathematical Programming Computation*, 3(4):349–390, 2011.

[43] A. Løkketangen. Heuristics for 0-1 mixed integer programming. *Handbook of Applied Optimization*, 2002.

[44] A. Løkketangen and F. Glover. Solving zero/one mixed integer programming problems using tabu search. *European Journal of Operations Research*, 106:624–658, 1998.

[45] H. Mittelmann. Decision tree for optimization software: Benchmarks for optimization software. `http://plato.asu.edu/bench.html`.

[46] G. Nannicini and P. Belotti. Rounding-based heuristics for nonconvex MINLPs. *Mathematical Programming Computation*, 4(1):1–31, 2012.

[47] G. Nannicini, P. Belotti, and L. Liberti. A local branching heuristic for MINLPs. *ArXiv e-prints*, 2008.

[48] E. Rothberg. An evolutionary algorithm for polishing mixed integer programming solutions. *INFORMS Journal on Computing*, 19(4):534–541, 2007.

[49] M. W. P. Savelsbergh. Preprocessing and probing techniques for mixed integer programming problems. *ORSA Journal on Computing*, 6:445–454, 1994.

[50] M. Tawarmalani and N. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563–591, 2004.

[51] S. Vigerske. *Decomposition in Multistage Stochastic Programming and a Constraint Integer Programming Approach to Mixed-Integer Nonlinear Programming*. PhD thesis, Humboldt-Universität zu Berlin, 2012.

[52] A. Wächter and L. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[53] C. Wallace. ZI round, a MIP rounding heuristic. *Journal of Heuristics*, 16(5):715–722, 2010.

[54] L. A. Wolsey. Faces for a linear inequality in 0-1 variables. *Mathematical Programming*, 8:165–178, 1975.

[55] R. Wunderling. *Paralleler und objektorientierter Simplex-Algorithmus*. PhD thesis, Technische Universität Berlin, 1996.

[56] CBC user guide – COIN-OR. http://www.coin-or.org/Cbc.

[57] CppAD. A Package for Differentiation of C++ Algorithms. http://www.coin-or.org/CppAD/.

[58] Ipopt (Interior Point OPTimizer). http://www.coin-or.org/Ipopt/.

[59] LindoGlobal. Lindo Systems, Inc. http://www.lindo.com.

[60] SBB. ARKI Consulting & Development A/S and GAMS Inc. http://www.gams.com/solvers/solvers.htm#SBB.

[61] SCIP. Solving Constraint Integer Programs. http://scip.zib.de/.

[62] SoPlex. An open source LP solver implementing the revised simplex algorithm. http://soplex.zib.de/.

List of Tables

Table 4: Computing optimal roundings for MIPLIB instances, after cuts

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
10teams	86.9	92.6	0.8	0.0	0	–	–
30n20b8	97.3	98.1	42.0	0.0	0	–	–
a1c1s1	18.8	7.9	6.6	limit	404552	13209.1836	271570
acc-tight5	58.8	78.1	6.0	0.7	0	–	–
aflow30a	78.9	80.4	4.4	3.6	3777	1158*	357
aflow40b	91.8	92.6	13.1	43.1	67215	1179	19497
air04	96.0	99.6	7.0	0.0	0	–	–
air05	96.1	98.9	2.4	0.0	0	–	–
app1-2	96.1	48.5	52.8	115.7	598	–	–
arki001	85.7	68.5	1.5	0.2	1	–	–
ash608gpia-3col	28.0	53.5	22.1	0.0	0	–	–
atlanta-ip	88.6	97.0	59.8	2.6	27	98.0096	22
bab5	97.2	99.4	56.3	0.1	0	–	–
beasleyC3	63.7	73.4	3.5	1.5	779	789	428
bell3a	96.2	90.2	0.0	0.0	1	878430.316*	1
bell5	72.3	77.5	0.1	0.0	0	–	–
biella1	90.5	92.0	5.6	limit	1439186	3278480.58	904043
bienst2	0.0	0.0	1.1	1634.4	459071	54.6*	49778
binkar10_1	48.8	48.7	0.9	270.9	407041	6746.64	89429
blend2	90.6	90.8	0.3	0.1	35	7.599*	22
bley_xl1	27.5	64.2	226.5	3.9	18	190*	18
bnatt350	50.4	66.3	4.2	0.0	0	–	–
cap6000	99.9	100.0	1.8	0.0	1	-2443599	1
core2536-691	94.6	94.8	11.5	3289.1	544659	695	10446
cov1075	25.0	25.0	0.9	35.0	10410	20*	506
csched010	88.7	84.3	2.9	1.0	38	–	–
dano3mip	67.4	64.6	30.6	limit	14384	762.75	2737
danoint	5.4	0.6	1.2	450.3	109479	65.6667*	5463
dcmulti	29.7	21.9	0.7	0.4	180	188186.5	68
dfn-gwin-UUM	38.9	13.4	0.5	819.3	307149	39920	4343
disctom	97.5	99.5	2.1	0.0	0	–	–
ds	99.0	99.4	105.5	0.6	0	–	–
dsbmip	84.5	21.5	0.8	0.2	34	-305.1982*	34
egout	85.7	85.7	0.0	0.0	1	568.1007*	1
eil33-2	98.5	99.7	6.3	0.0	0	–	–
eilB101	88.9	99.0	13.5	0.1	0	–	–
enigma	83.0	92.0	0.0	0.0	0	–	–
enlight13	66.6	96.2	0.2	0.0	0	–	–
enlight14	68.4	95.9	0.3	0.0	0	–	–
fast0507	99.5	99.5	14.3	14.4	10302	177	4218
fiber	91.9	95.0	0.9	0.1	78	411151.82	48
fixnet6	88.6	82.3	1.1	0.4	32	3997	26
flugpl	11.1	35.7	0.0	0.0	0	–	–
gesa2	88.0	82.4	1.1	0.0	5	25780031.4*	3
gesa2-o	92.9	88.9	1.0	0.0	5	25780031.4*	3
gesa3	78.6	82.0	1.3	0.0	36	27991430.1	33
gesa3_o	85.0	85.6	1.2	0.0	19	27991430.1	17
glass4	70.8	74.4	0.3	1.6	2622	2.2666856e+09	2491
gmu-35-40	93.5	93.7	0.5	0.1	61	-2399398.21	57
gt2	90.8	100.0	0.0	0.0	1	21166*	1
harp2	91.1	98.3	0.8	0.0	0	–	–
iis-100-0-cov	0.0	0.0	2.6	1700.5	120842	29*	30
iis-bupa-cov	57.8	57.8	8.8	3819.8	537082	36*	1634
iis-pima-cov	82.3	82.3	17.9	54.6	12823	33*	4545
khb05250	66.7	32.6	0.3	0.1	7	106940226*	4

Table 4 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
l152lav	97.2	99.4	0.1	0.0	0	–	–
lectsched-4-obj	28.7	31.0	6.8	0.0	0	–	–
liu	49.0	46.2	10.8	limit	6040599	3418	4613091
lseu	74.4	77.9	0.1	0.1	22	1148	18
m100n500k4r1	73.2	73.2	0.4	0.8	848	-22	180
macrophage	43.5	45.8	2.3	0.0	0	–	–
map18	63.6	76.6	48.8	128.7	2896	-847*	711
map20	63.6	75.7	38.9	104.5	2408	-922*	888
markshare1	76.0	76.0	0.0	0.0	107	142	59
markshare2	78.3	78.3	0.1	0.0	101	131	94
mas74	91.3	90.7	0.2	0.4	90	14343.468	67
mas76	91.9	91.3	0.2	0.3	42	40560.0541	35
mcsched	15.9	18.4	3.0	limit	1721772	213768	54512
mik-250-1-100-1	62.4	62.2	0.2	0.2	172	-66729*	172
mine-90-10	20.6	27.8	4.2	limit	2667271	-784302338*	2445697
misc03	78.3	99.3	0.2	0.0	0	–	–
misc06	90.2	38.4	0.2	0.1	19	12850.8607*	17
misc07	82.8	94.0	0.3	0.0	0	–	–
mitre	99.6	100.0	4.8	0.0	1	115155*	1
mkc	92.6	93.5	2.9	0.3	389	-539.866	160
mod008	94.4	94.4	0.7	0.1	19	309	4
mod010	98.6	100.0	0.3	0.0	0	–	–
mod011	53.1	12.6	7.2	64.1	387	-54219145.9	129
modglob	60.2	56.8	0.2	1.3	5795	20799458.8	4360
momentum1	76.7	73.0	11.8	0.2	0	–	–
momentum2	74.8	76.5	50.9	0.7	0	–	–
momentum3	78.4	77.1	1034.8	0.5	0	–	–
msc98-ip	82.0	85.5	145.8	0.1	0	–	–
mspp16	99.0	99.1	1202.2	13.1	0	–	–
mzzv11	83.4	82.9	74.8	0.0	0	–	–
mzzv42z	86.5	86.1	75.4	0.1	0	–	–
n3div36	99.9	99.9	6.7	0.1	1	151600	1
n3seq24	99.6	99.7	82.8	63.1	24054	68000	3536
n4-3	56.9	10.0	2.8	limit	415575	9010	112840
neos13	78.6	78.1	26.8	limit	75103	-65.6552	51090
neos18	70.8	78.1	0.9	0.0	0	–	–
neos-1109824	94.5	97.0	3.2	0.0	0	–	–
neos-1337307	45.1	45.2	4.7	limit	767115	-202133	4868
neos-1396125	45.0	48.0	3.5	11.4	2026	3000.0553*	1867
neos-1601936	80.8	77.1	7.9	limit	252812	–	–
neos-476283	99.0	93.0	147.4	3.0	130	406.8123	71
neos-686190	96.0	98.3	1.3	0.0	0	–	–
neos-849702	70.8	80.0	1.6	0.1	0	–	–
neos-916792	87.1	89.3	13.1	0.1	0	–	–
neos-934278	76.8	75.1	49.9	limit	105271	1332	9576
net12	41.8	56.3	31.7	0.1	0	–	–
netdiversion	96.1	99.9	301.9	1.1	0	–	–
newdano	0.0	0.0	2.9	limit	1160686	66.5	774340
noswot	47.4	64.2	0.1	0.0	0	–	–
ns1208400	78.2	82.5	6.2	0.1	0	–	–
ns1688347	99.4	99.9	20.1	0.0	0	–	–
ns1758913	91.4	92.1	5385.0	6.4	5	-457.7183	5
ns1766074	77.8	87.0	0.1	0.0	1	–	–
ns1830653	57.8	72.8	4.6	0.1	0	–	–
nsrand-ipx	98.3	98.4	19.7	569.0	2061551	55360	31084
nw04	100.0	100.0	14.1	0.4	0	–	–
opm2-z7-s2	9.9	10.1	10.9	limit	52398	-10271	50719
opt1217	95.2	96.9	0.3	0.0	1	-16*	1

Table 4 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
p0201	63.1	92.8	0.4	0.0	1	7805	1
p0282	71.5	72.5	0.3	0.1	1	258411*	1
p0548	96.6	100.0	0.2	0.0	1	8763	1
p2756	98.9	99.6	1.1	0.0	1	3152	1
pg5_34	97.0	46.0	3.8	1.5	7	-14287.7021	4
pigeon-10	44.6	77.2	1.3	7.7	27538	–	–
pk1	72.7	46.5	0.0	0.2	460	29	376
pp08a	46.9	33.8	0.3	0.6	319	7360	148
pp08aCUTS	48.4	32.1	0.2	0.6	434	7370	405
protfold	65.8	88.4	3.8	0.2	0	–	–
pw-myciel4	58.4	60.4	7.6	0.0	0	–	–
qiu	25.0	25.0	0.2	47.6	23791	-132.8731*	1149
qnet1	92.0	95.1	0.9	0.0	1	21237.6552	1
qnet1_o	91.7	95.0	1.1	0.1	261	22600.83	168
rail507	99.5	99.5	14.8	41.2	23871	178	230
ran16x16	71.1	71.5	1.1	138.7	464014	3846	4332
rd-rplusc-21	55.5	66.0	57.7	4.3	415	–	–
reblock67	17.6	26.6	3.7	limit	5552244	-34629815.5	700261
rentacar	75.0	7.6	1.2	0.6	9	30356761*	6
rgn	96.0	54.9	0.2	0.0	1	82.2*	1
rmatr100-p10	49.0	49.2	2.8	16.5	686	424	322
rmatr100-p5	63.0	63.6	4.1	13.0	258	976*	118
rmine6	65.5	67.0	8.2	5266.1	4687190	-457.1727	811719
rocII-4-11	81.6	88.4	18.4	0.0	0	–	–
rococoC10-001000	82.1	85.8	2.7	33.5	42970	12067	4679
roll3000	65.2	78.3	2.7	0.4	94	14193	12
rout	83.2	93.7	0.6	0.0	0	–	–
satellites1-25	89.2	99.4	68.2	0.0	0	–	–
set1ch	96.2	90.2	0.7	0.0	3	54537.75*	2
seymour	52.7	55.5	15.1	limit	1067621	427	917345
sp98ic	99.3	99.3	4.8	12.2	37885	469766019	12687
sp98ir	93.9	96.0	3.2	0.0	0	–	–
stein27	11.1	11.1	0.0	0.3	1202	18*	50
stein45	17.8	17.8	0.2	0.9	3597	30*	313
swath	99.2	98.3	3.4	0.0	0	–	–
t1717	99.2	99.4	27.3	0.4	0	–	–
tanglegram1	99.1	99.1	14.4	0.2	0	–	–
tanglegram2	96.4	96.7	1.3	0.0	0	–	–
timtab1	14.4	15.9	0.8	8.3	16082	827609	4701
timtab2	12.6	14.7	1.7	2.4	151	–	–
tr12-30	73.6	50.7	1.5	408.3	909211	131438	17370
triptim1	87.0	99.5	127.3	0.2	0	–	–
unitcal_7	81.6	59.7	63.9	2.8	1	–	–
vpm2	55.4	50.8	0.4	0.4	336	13.75*	301
vpphard	97.6	98.1	28.4	0.6	0	–	–
zib54-UUE	17.5	21.5	2.6	limit	1126102	10334015.8*	392023

Table 5: Computing optimal roundings for MIPLIB instances, before cuts

	% Vars Fixed			RENS			
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
10teams	90.1	92.5	0.4	0.0	0	–	–
30n20b8	98.1	98.8	2.2	0.0	0	–	–
a1c1s1	15.6	10.4	3.6	limit	2174731	–	–
acc-tight5	56.8	84.4	2.0	0.1	0	–	–
aflow30a	92.6	97.9	0.2	0.0	0	–	–
aflow40b	97.2	98.0	1.0	0.0	0	–	–
air04	96.1	98.2	6.2	0.0	0	–	–
air05	96.4	98.7	1.8	0.0	0	–	–
app1-2	96.7	48.9	14.9	60.3	492	-23	492
arki001	84.9	72.6	0.5	0.0	0	–	–
ash608gpia-3col	33.4	67.3	3.3	0.0	0	–	–
atlanta-ip	88.9	97.3	30.2	1.8	196	99.0098	195
bab5	98.8	99.1	29.0	0.1	0	–	–
beasleyC3	82.4	100.0	0.1	0.0	1	945	1
bell3a	84.6	80.4	0.0	0.0	13	878651.068	12
bell5	70.2	87.5	0.0	0.0	1	9082700.02	1
biella1	90.5	92.0	5.2	limit	1251065	3253217.92	682395
bienst2	0.0	0.0	0.3	1133.6	514667	54.6*	248177
binkar10_1	77.6	77.6	0.1	1.1	2688	6796.71	1565
blend2	97.4	99.3	0.0	0.0	0	–	–
bley_xl1	47.4	82.2	171.8	0.4	11	210	11
bnatt350	58.9	59.4	1.3	0.0	0	–	–
cap6000	100.0	100.0	0.6	0.0	1	-2442801	1
core2536-691	94.6	94.8	11.2	5427.0	951274	695	30373
cov1075	0.0	0.0	0.6	limit	1622177	20*	184
csched010	94.2	91.6	0.3	0.0	1	–	–
dano3mip	77.5	74.4	22.1	limit	25874	761.9286	118
danoint	7.1	0.8	0.6	152.0	50018	65.6667*	40513
dcmulti	35.1	41.4	0.1	9.4	40800	188182*	12687
dfn-gwin-UUM	50.0	4.8	0.1	54.1	108721	41040	20493
disctom	97.5	99.5	1.8	0.0	0	–	–
ds	99.2	99.5	27.0	0.5	0	–	–
dsbmip	74.1	20.6	0.5	0.2	7	–	–
egout	71.4	100.0	0.0	0.0	1	625.3192	1
eil33-2	99.3	99.9	3.6	0.0	0	–	–
eilB101	96.8	97.8	1.6	0.0	0	–	–
enigma	88.0	99.0	0.0	0.0	0	–	–
enlight13	99.1	99.1	0.0	0.0	0	–	–
enlight14	99.2	99.2	0.0	0.0	0	–	–
fast0507	99.5	99.5	13.2	15.6	12207	177	5197
fiber	96.2	100.0	0.0	0.0	0	–	–
fixnet6	96.8	90.6	0.0	0.0	3	4435	3
flugpl	11.1	21.4	0.0	0.0	0	–	–
gesa2	89.7	91.6	0.2	0.0	5	26038337.6	5
gesa2-o	89.9	96.0	0.2	0.0	6	26038337.6	5
gesa3	81.5	88.1	0.2	0.0	29	27991430.1	24
gesa3_o	84.6	90.6	0.2	0.0	29	27991430.1	24
glass4	75.8	83.9	0.1	0.1	49	–	–
gmu-35-40	98.3	98.6	0.3	0.0	0	–	–
gt2	91.3	96.5	0.0	0.0	0	–	–
harp2	97.8	99.3	0.1	0.0	0	–	–
iis-100-0-cov	0.0	0.0	0.6	2902.6	186105	29*	47
iis-bupa-cov	55.1	55.1	1.3	6150.5	745491	36*	1989
iis-pima-cov	82.1	82.1	1.8	50.7	11558	33*	1363
khb05250	20.8	4.3	0.0	1.6	3364	106940226*	87

Table 5 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
l152lav	97.2	99.9	0.2	0.0	0	–	–
lectsched-4-obj	78.2	78.8	1.5	0.0	0	–	–
liu	51.4	48.4	36.3	limit	8755631	4762	2865
lseu	90.7	100.0	0.0	0.0	0	–	–
m100n500k4r1	80.0	80.0	-0.0	0.5	650	-21	102
macrophage	70.0	70.5	0.1	0.0	0	–	–
map18	58.5	71.5	33.3	3299.9	61952	-847*	52
map20	66.1	80.4	27.0	404.2	17845	-922*	617
markshare1	88.0	92.0	0.0	0.0	1	204	1
markshare2	88.3	88.3	0.0	0.0	3	131	2
mas74	91.9	91.3	0.0	0.0	58	14372.8713	20
mas76	92.6	92.0	0.0	0.0	21	40560.0541	12
mcsched	15.8	18.2	0.9	limit	1966420	214792	1088285
mik-250-1-100-1	60.0	59.8	0.1	0.0	32	0	31
mine-90-10	20.6	27.8	3.8	limit	2556662	-782117611	1315502
misc03	87.0	97.1	0.1	0.0	0	–	–
misc06	92.9	39.0	0.1	0.1	43	12854.0023	33
misc07	91.4	98.3	0.2	0.0	0	–	–
mitre	99.6	100.0	4.5	0.0	1	116745	1
mkc	97.6	99.1	1.1	0.0	1	-284.55	1
mod008	98.4	100.0	0.0	0.0	1	308	1
mod010	98.4	99.8	0.4	0.0	0	–	–
mod011	83.3	21.2	0.6	1.6	153	-53656254.1	50
modglob	69.4	76.6	0.1	0.0	174	20784597.9	174
momentum1	80.3	78.6	5.1	155.3	76443	109169.397	19330
momentum2	78.9	83.1	26.2	0.3	0	–	–
momentum3	77.3	78.4	497.4	0.6	0	–	–
msc98-ip	86.8	89.4	6.6	0.1	0	–	–
mspp16	99.9	100.0	1001.5	13.0	0	–	–
mzzv11	86.4	85.7	51.5	0.0	0	–	–
mzzv42z	88.2	87.8	51.9	0.0	0	–	–
n3div36	99.9	99.9	2.0	0.1	3	149800	2
n3seq24	99.8	99.9	23.0	1.5	0	–	–
n4-3	74.1	31.8	0.1	359.6	215073	9395	12131
neos13	78.2	77.7	12.2	39.6	267	-66.8793	267
neos18	71.4	71.7	0.3	0.0	0	–	–
neos-1109824	96.8	99.9	1.1	0.0	0	–	–
neos-1337307	50.0	50.1	2.4	742.6	154344	-202143	12623
neos-1396125	47.3	53.1	1.1	2.2	510	3000.0556*	489
neos-1601936	83.7	77.2	6.5	0.0	0	–	–
neos-476283	99.0	93.0	141.6	2.7	121	406.8123	74
neos-686190	96.9	99.0	0.2	0.0	0	–	–
neos-849702	74.5	80.9	1.2	0.0	0	–	–
neos-916792	87.1	89.3	1.1	0.1	0	–	–
neos-934278	79.5	78.0	18.6	limit	215616	346	201165
net12	60.4	79.8	7.9	0.1	0	–	–
netdiversion	96.5	100.0	199.6	1.0	0	–	–
newdano	3.6	0.4	0.4	1900.2	1332691	66.8333	800380
noswot	50.5	47.5	0.0	0.0	0	–	–
ns1208400	84.1	87.6	2.3	0.0	0	–	–
ns1688347	65.8	77.8	8.0	0.1	0	–	–
ns1758913	97.4	98.7	1437.2	1.3	41	-862.2649	37
ns1766074	77.8	86.0	0.0	0.0	7	–	–
ns1830653	57.8	50.3	1.2	0.0	0	–	–
nsrand-ipx	99.0	99.2	1.1	0.2	1381	61760	109
nw04	100.0	100.0	10.4	0.4	0	–	–
opm2-z7-s2	9.9	10.1	7.6	limit	52408	-10271	50719
opt1217	96.2	98.8	0.1	0.0	13	-16*	13

Table 5 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
p0201	78.5	98.5	0.1	0.0	0	–	–
p0282	96.0	100.0	0.0	0.0	1	320465	1
p0548	91.7	97.8	0.1	0.0	0	–	–
p2756	95.6	98.8	0.3	0.0	0	–	–
pg5_34	12.0	0.5	28.3	limit	5836732	-14232.4589	1862706
pigeon-10	69.5	99.3	0.1	0.0	0	–	–
pk1	72.7	46.5	0.0	0.1	402	29	247
pp08a	17.2	9.4	125.6	limit	33411470	7360	395800
pp08aCUTS	28.1	16.5	0.1	134.2	557900	7350*	29979
protfold	72.2	90.0	2.0	0.1	0	–	–
pw-myciel4	46.0	56.9	1.0	0.0	0	–	–
qiu	25.0	25.0	0.2	47.5	23791	-132.8731*	1149
qnet1	96.3	99.3	0.2	0.0	1	21396.52	1
qnet1_o	99.2	100.0	0.1	0.0	1	28462.14	1
rail507	99.5	99.5	13.7	100.5	66744	178	341
ran16x16	92.2	100.0	0.0	0.0	1	4333	1
rd-rplusc-21	77.9	78.8	46.0	0.2	0	–	–
reblock67	17.6	26.6	3.1	limit	6367150	-34629815.5	540746
rentacar	70.8	8.4	0.8	0.6	15	30356761*	13
rgn	85.0	48.6	0.1	0.0	109	82.2*	9
rmatr100-p10	49.0	49.2	2.6	16.4	686	424	322
rmatr100-p5	63.0	63.6	3.9	13.0	258	976*	118
rmine6	64.7	66.9	2.3	2730.2	2638869	-457.1727	1416590
rocII-4-11	85.9	89.6	14.3	0.0	0	–	–
rococoC10-001000	93.4	100.0	0.3	0.0	1	23730	1
roll3000	67.5	72.3	1.1	0.0	0	–	–
rout	88.9	93.9	0.2	0.0	0	–	–
satellites1-25	90.2	98.8	35.6	0.0	0	–	–
set1ch	45.1	44.6	1077.1	limit	41287842	–	–
seymour	48.3	48.9	3.4	limit	700335	428	607424
sp98ic	99.3	99.3	2.1	19.2	70178	469766019	478
sp98ir	94.3	97.1	2.3	0.0	0	–	–
stein27	22.2	22.2	0.1	0.0	224	18*	18
stein45	22.2	22.2	0.0	0.4	1507	30*	510
swath	99.3	98.7	2.6	0.0	0	–	–
t1717	99.2	99.4	7.4	0.4	0	–	–
tanglegram1	99.2	99.2	2.0	0.2	0	–	–
tanglegram2	96.9	97.1	0.3	0.0	0	–	–
timtab1	36.3	51.7	0.1	0.0	1	–	–
timtab2	22.8	42.8	0.0	0.1	1	–	–
tr12-30	7.4	6.2	70.5	limit	11643684	–	–
triptim1	87.1	99.7	103.8	0.2	0	–	–
unitcal_7	80.1	48.3	29.3	0.1	0	–	–
vpm2	63.9	77.3	0.0	0.0	14	15.25	12
vpphard	97.8	98.0	17.1	0.2	0	–	–
zib54-UUE	26.3	25.4	3.4	limit	1588278	10334015.8*	64873

Table 6: Computing optimal roundings for MIQCP instances, using LP solution, after cuts

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
10bar2	77.3	76.0	0.2	0.0	14	2691.7039	13
25bar	83.9	49.2	0.1	0.1	20	–	–
classical_200_0	92.0	59.8	1.6	1.7	31	-0.0848	26
classical_200_1	90.5	59.0	1.5	6.3	313	-0.097	195
classical_20_0	60.0	25.0	0.1	0.1	15	-0.0686	11
classical_20_1	55.0	23.3	0.0	0.3	40	-0.0698	16
classical_50_0	72.0	42.7	0.4	0.9	116	-0.0818	99
classical_50_1	82.0	49.3	0.2	0.6	69	-0.0737	6
clay0203m	20.0	14.8	0.1	0.0	55	41573.0147*	10
clay0205m	35.6	32.0	0.1	0.4	341	8672.5	184
clay0303m	21.1	22.6	0.1	0.1	61	41573.0276	53
clay0305m	29.4	25.9	0.2	0.5	724	8488.3117	716
du-opt5	54.5	5.3	0.1	0.2	29	–	–
du-opt	30.8	0.0	0.1	1.8	335	–	–
ex1263	69.0	69.2	0.2	0.0	1	28.3	1
ex1266	81.7	97.6	0.2	0.0	1	21.3	1
fac3	0.0	0.0	0.1	0.1	25	31982309.8*	13
feedtray2	41.7	29.7	24.4	limit	2358782	–	–
ibell3a	88.3	85.2	0.1	0.0	1	878785.031*	1
icvxqp1	99.7	100.0	454.9	0.6	1	914601	1
ilaser0	0.0	5.7	1.2	limit	237295	–	–
imod011	71.1	23.4	233.9	6341.0	345627	362636789	333111
iportfolio	80.1	64.5	4.3	283.9	26983	–	–
isqp	62.0	2.4	3.9	limit	97291	–	–
itointqor	86.0	94.1	0.0	0.0	1	53624064.4	1
ivalues	68.8	40.9	0.8	0.0	1	9026.4463	1
meanvarx	83.3	66.7	0.0	0.0	5	14.3692*	4
netmod_dol1	16.7	16.7	1.3	4622.7	82905	-0.5562	99
netmod_dol2	47.4	36.1	1.9	774.4	24560	-0.545	3448
netmod_kar1	0.0	0.0	0.3	1.9	327	-0.4198*	8
netmod_kar2	0.0	0.0	0.3	1.8	327	-0.4198*	8
nous1	0.0	0.0	290.7	limit	6203637	–	–
nous2	0.0	0.0	393.2	limit	5777698	–	–
nuclear14a	83.5	63.6	16.7	limit	94439	–	–
nuclear14b	92.7	71.7	2.0	3.7	111	–	–
nvs19	0.0	0.0	0.0	0.0	9	-1098.4*	9
nvs23	0.0	0.0	0.1	0.0	1	-1124.2	1
product2	81.2	26.9	162.5	limit	5890550	–	–
product	67.4	50.4	0.7	389.8	650612	-2130.6323	255299
robust_100_0	88.1	41.9	1.1	0.7	23	-0.0888	12
robust_100_1	86.1	41.2	0.9	1.6	123	-0.0525	63
robust_200_0	89.6	43.8	2.0	1.9	121	-0.0944	20
robust_20_0	85.7	32.5	0.1	0.0	5	-0.0759	2
robust_50_0	82.4	37.4	0.5	0.3	38	-0.0671	16
robust_50_1	82.4	37.4	0.5	0.4	50	-0.0714	34
shortfall_100_0	76.2	35.9	0.9	0.9	45	-1.0737	36
shortfall_100_1	83.2	39.4	1.1	0.8	33	-1.0657	32
shortfall_200_0	88.6	43.2	2.5	3.0	45	-1.0803	45
shortfall_20_0	71.4	25.0	0.0	0.1	11	-1.0811	10
shortfall_50_0	72.5	31.9	0.4	0.7	27	-1.0799	21
shortfall_50_1	78.4	34.8	0.3	0.5	21	-1.0806	18
SLay05H	67.5	60.6	0.3	0.3	33	24809.6753	31
SLay05M	55.0	43.7	0.1	0.1	33	33732.8607	9
SLay07M	71.4	48.9	0.1	0.4	63	73105.8847	33
SLay10H	41.1	38.2	19.6	limit	754162	131656.989	105106

Table 6 continued

Instance	% Vars Fixed Int	All	RENS TimeS	TimeR	NodesR	Solution	Found At
SLay10M	68.3	51.9	0.4	1.9	415	185502.124	392
space25a	96.7	82.5	0.2	0.0	5	–	–
space25	94.6	80.1	0.4	0.7	8407	–	–
spectra2	80.0	70.6	0.4	0.1	26	13.9783*	14
tln12	48.2	52.2	0.2	0.0	0	–	–
tln5	74.3	77.1	0.0	0.0	0	–	–
tln6	64.6	68.8	0.1	0.0	0	–	–
tln7	42.9	49.2	0.1	0.0	0	–	–
tloss	69.6	82.6	0.0	0.0	0	–	–
tltr	25.5	39.3	0.1	0.0	0	–	–
uflquad-15-60	0.0	0.0	2.8	2679.7	1052	1063.1929*	237
uflquad-20-50	0.0	0.0	25.1	limit	128	474.9019	64
uflquad-40-80	97.5	85.1	1.7	limit	2	–	–
util	91.7	46.9	0.0	0.0	10	1000.9676	10
waste	97.5	91.5	0.4	0.1	426	692.7824	291

Table 7: Computing optimal roundings for MIQCP instances, using NLP solution, after cuts

	% Vars Fixed			RENS			
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
10bar2	0.0	0.0	0.2	5.1	2678	1960.4104	2571
25bar	23.0	12.5	0.2	5.2	1199	400.3246	1192
classical_200_0	0.0	0.0	21.5	limit	157452	-0.1042	19694
classical_200_1	0.0	0.0	24.2	limit	184429	-0.1092	67634
classical_20_0	0.0	0.0	0.1	1.0	1354	-0.0823*	834
classical_20_1	0.0	0.0	-0.0	2.4	1835	-0.0757*	1747
classical_50_0	0.0	0.0	0.4	784.7	199803	-0.0907*	133471
classical_50_1	0.0	0.0	0.2	61.8	20511	-0.0948*	17026
clay0203m	0.0	0.0	0.1	0.1	110	41573.0265*	95
clay0205m	0.0	0.0	0.2	3.0	10442	8092.5*	1759
clay0303m	0.0	0.0	0.1	0.2	167	26669.0752	156
clay0305m	0.0	0.0	0.2	6.1	17597	8092.5*	1579
du-opt5	45.5	5.3	0.1	0.1	25	–	–
du-opt	0.0	0.0	0.1	34.0	6827	–	–
ex1263	45.1	52.7	0.3	0.2	70	20.3	49
ex1266	65.9	69.6	0.2	0.1	40	16.3*	40
fac3	8.3	1.5	1.0	0.0	23	31982309.8*	13
feedtray2	0.0	0.0	0.1	247.5	96287	0*	96287
ibell3a	60.0	82.8	0.1	0.0	1	879009.262	1
icvxqp1	97.6	98.1	580.3	0.6	1	375878	1
ilaser0	0.0	7.7	1.0	0.0	0	–	–
imod011	–	–	1346.6	–	–	–	–
iportfolio	0.0	0.0	6.9	limit	276015	–	–
isqp	0.0	0.0	331.7	limit	800472	–	–
itointqor	0.0	0.0	60.4	limit	31848641	-1145.95	30734174
ivalues	51.5	6.4	0.7	45.2	262102	-1.1657*	20497
meanvarx	58.3	56.7	0.1	0.0	5	14.3692*	4
netmod_dol1	0.0	0.0	13.8	limit	70283	-0.56*	197
netmod_dol2	24.4	24.1	4.7	12.3	365	-0.5208	216
netmod_kar1	0.0	0.0	0.4	1.9	327	-0.4198*	8
netmod_kar2	0.0	0.0	0.2	1.9	327	-0.4198*	8
nous1	0.0	0.0	295.1	limit	6189939	–	–
nous2	0.0	0.0	401.9	limit	5775976	–	–
nuclear14a	0.0	0.0	18.4	limit	98876	–	–
nuclear14b	0.0	0.0	39.2	limit	122109	–	–
nvs19	0.0	0.0	0.0	0.1	53	-1098.2	52
nvs23	0.0	0.0	0.0	0.2	75	-1124.8	73
product2	9.4	11.5	226.2	limit	5344387	–	–
product	67.4	41.6	159.1	limit	3714246	–	–
robust_100_0	0.0	0.0	36.0	limit	643608	-0.0964	432103
robust_100_1	0.0	0.0	28.5	limit	749339	-0.0716	500948
robust_200_0	0.0	0.0	20.3	limit	194374	-0.1359	57193
robust_20_0	0.0	0.0	0.1	0.1	11	-0.0798*	6
robust_50_0	0.0	0.0	0.6	1.2	270	-0.0861*	156
robust_50_1	0.0	0.0	0.3	12.3	3064	-0.0857*	754
shortfall_100_0	0.0	0.0	51.8	limit	418270	-1.1023	57765
shortfall_100_1	0.0	0.0	60.5	limit	459850	-1.094	168978
shortfall_200_0	0.0	0.0	32.9	limit	130390	-1.1096	10874
shortfall_20_0	0.0	0.0	0.1	0.6	624	-1.0905*	157
shortfall_50_0	0.0	0.0	74.0	limit	1248837	-1.095	930028
shortfall_50_1	0.0	0.0	0.5	1975.5	520190	-1.1018*	427638
SLay05H	0.0	0.0	0.2	7.0	3094	22664.678*	1400
SLay05M	0.0	0.0	0.1	1.7	878	22664.6781*	536
SLay07M	0.0	0.0	0.0	59.6	29886	64748.8243*	9877
SLay10H	0.0	0.0	18.5	limit	468624	130031.675	129100

Table 7 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
SLay10M	0.0	0.0	16.4	limit	834497	129771.879	740342
space25a	41.7	32.5	0.3	limit	6254	–	–
space25	41.7	34.4	0.5	limit	894	–	–
spectra2	80.0	70.6	0.5	0.1	26	13.9783*	14
tln12	2.4	0.0	0.5	0.0	0	–	–
tln5	22.9	40.0	0.1	0.0	0	–	–
tln6	18.8	35.4	0.1	0.0	0	–	–
tln7	19.0	31.7	0.1	0.0	0	–	–
tloss	69.6	82.6	0.1	0.0	0	–	–
tltr	27.7	73.2	0.1	0.0	0	–	–
uflquad-15-60	0.0	0.0	2.9	2701.2	1052	1063.1929*	237
uflquad-20-50	0.0	0.0	25.1	limit	128	474.9019	64
uflquad-40-80	0.0	0.0	3.0	limit	1083	–	–
util	0.0	0.0	0.0	0.1	455	999.5788*	224
waste	86.3	75.1	401.5	limit	13736796	–	–

Table 8: Computing optimal roundings for MINLP instances, using LP solution, after cuts

	% Vars Fixed			RENS			
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
beuster	76.5	40.2	118.1	7049.7	15735321	–	–
cecil_13	25.0	19.4	18.5	7010.4	6032779	-115599.148	6497
chp_partload	35.7	2.8	4.5	7158.7	13536	–	–
contvar	89.7	13.6	2.6	limit	89028	–	–
csched1	95.0	78.7	0.1	0.0	45	-29775.9885	45
csched2a	60.0	38.5	76.1	7059.7	5436390	-94800.4303	2043936
eg_all_s	28.6	53.0	589.3	6598.6	80557	–	–
eg_disc2_s	0.0	13.4	286.4	6970.9	22	–	–
eg_disc_s	50.0	36.6	316.1	6886.8	858	–	–
eg_int_s	0.0	14.3	501.2	6723.7	5	–	–
eniplac	30.4	26.2	0.1	0.1	151	-132117.083*	37
enpro48	80.4	77.3	0.1	15.5	111594	241150.752	111594
enpro48pb	79.3	71.4	0.0	1.1	4634	264032.12	4634
enpro56	67.1	56.0	0.2	17.7	147897	279702.866	147897
enpro56pb	65.7	53.6	0.1	5.4	41063	279704.1	41063
ex1233	20.0	7.2	1.6	limit	189292	–	–
ex1244	40.0	36.7	0.2	0.0	28	84035.1235	23
ex1252a	77.8	57.8	0.0	0.0	0	–	–
ex1252	71.4	55.4	0.1	1.2	5342	–	–
feedtray	42.9	1.2	68.9	7115.3	874824	–	–
fo7_2	19.0	9.8	0.1	3.3	14805	17.7493*	627
fo7_ar2_1	24.4	12.3	0.1	291.9	2381312	26.9425	2381312
fo7_ar25_1	36.6	18.5	0.1	0.4	755	25.6421	326
fo7_ar3_1	43.9	22.2	0.0	0.5	976	25.6421	316
fo7_ar4_1	29.3	14.8	0.1	2.3	9622	24.3794	4178
fo7_ar5_1	34.1	17.3	0.0	0.9	2147	19.6229	566
fo7	16.7	8.5	0.0	120.9	558800	30.6572	382347
fo8_ar2_1	36.4	20.8	0.2	2.1	6262	41.8507	3493
fo8_ar25_1	16.4	8.9	0.2	108.1	453566	28.0452*	84041
fo8_ar3_1	38.2	20.8	0.1	2.9	8133	–	–
fo8_ar4_1	30.9	16.8	0.1	146.9	975930	32.5005	968495
fo8_ar5_1	30.9	16.8	0.2	6.1	21065	24.4077	3434
fo8	21.4	11.8	0.2	592.2	2279417	37.2612	216937
fo9_ar2_1	23.9	13.8	0.1	1.7	5290	45.8141	3577
fo9_ar25_1	35.2	20.3	0.2	15.5	46324	32.6795	23480
fo9_ar3_1	22.5	13.0	0.1	598.5	1658625	37.5937	8325
fo9_ar4_1	25.4	14.6	0.2	879.3	2599259	37.1576	29588
fo9_ar5_1	28.2	16.3	0.2	65.9	196069	26.9217	134598
fo9	19.4	11.3	34.7	7053.4	20841677	34.6228	6480181
fuzzy	71.8	42.6	86.8	7126.7	4547053	–	–
gasnet	50.0	23.6	0.1	limit	3063	–	–
ghg_1veh	0.0	0.0	386.4	7108.6	6426635	–	–
ghg_2veh	18.8	7.6	109.5	7130.6	1936310	–	–
ghg_3veh	51.4	21.3	37.6	7163.3	1587865	–	–
hda	28.6	18.0	6.6	limit	588838	–	–
m6	3.3	1.6	0.0	2.2	11390	82.2569*	3883
m7_ar2_1	13.3	5.9	0.1	1.5	10467	195.035	9794
m7_ar25_1	18.8	8.6	0.1	0.2	443	143.585*	204
m7_ar3_1	34.2	17.1	0.1	0.5	772	152.5792	330
m7_ar4_1	34.1	17.7	0.2	0.3	730	130.46	287
m7_ar5_1	26.8	13.9	0.1	1.0	4354	148.6199	1740
m7	33.3	17.5	0.1	0.1	341	126.4312	196
mbtd	–	–	limit	–	–	–	–
no7_ar2_1	36.6	17.2	0.2	0.8	1772	150.7814	740
no7_ar25_1	26.8	12.6	0.0	2.7	9032	107.8663	7186

Table 8 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
no7_ar3_1	26.8	12.6	0.2	1.2	3223	119.3432	2131
no7_ar4_1	43.9	20.7	0.0	1.1	3492	117.8947	2278
no7_ar5_1	24.4	11.5	0.1	28.6	104622	100.8113	10082
nvs09	60.0	55.0	534.1	6969.9	77479321	-11.1518	15924294
nvs20	20.0	6.1	0.0	1.2	1948	230.9221*	1580
o7_2	31.0	14.4	0.1	8.6	33559	129.4105	2060
o7_ar2_1	31.7	14.6	0.2	2.8	10741	140.4119*	188
o7_ar25_1	36.6	16.9	0.1	29.0	182612	143.1372	182612
o7_ar3_1	26.8	12.4	0.1	10.9	34069	–	–
o7_ar4_1	26.8	12.4	0.1	7.2	27844	143.8912	24195
o7_ar5_1	46.3	21.3	0.1	31.0	213317	135.7148	213317
o7	19.0	8.9	0.1	428.8	1812739	139.4551	207218
o8_ar4_1	32.7	15.4	0.2	28.0	65139	–	–
o9_ar4_1	39.4	20.4	0.1	119.5	311859	–	–
oil2	50.0	0.5	1.6	limit	1205253	–	–
oil	57.9	8.3	24.3	7177.6	165976	–	–
parallel	20.0	14.7	8.1	7184.6	899801	924.225	834864
pump	77.8	57.8	0.0	0.0	0	–	–
risk2b	66.7	5.6	0.2	0.0	11	-55.8761*	9
spring	91.7	67.9	0.0	0.0	0	–	–
st_e32	88.9	29.7	0.1	0.0	3	–	–
stockcycle	86.8	91.3	0.8	0.0	51	334280.188	46
super1	83.9	10.0	1.2	0.0	0	–	–
super2	71.0	8.4	1.1	0.0	0	–	–
super3	67.6	8.6	1.2	0.0	0	–	–
super3t	35.1	6.2	8.8	7157.6	76873	–	–
synheat	20.0	8.0	17.7	limit	3475310	–	–
synthes1	0.0	0.0	0.0	0.0	5	6.0098*	4
synthes2	50.0	36.4	0.0	0.0	6	73.0353*	6
synthes3	42.9	29.4	0.1	0.0	11	68.0097*	10
tls12	93.7	81.0	1.7	0.0	0	–	–
tls4	55.3	53.2	0.2	0.2	417	11.5	338
tls5	64.1	64.0	0.5	0.4	2073	12.5	2043
tls6	86.1	83.1	0.3	0.0	0	–	–
tls7	90.7	64.9	0.5	0.0	0	–	–
water3	67.9	35.3	0.1	292.7	972217	907.0153	779595
waterful2	92.9	76.4	0.2	4.8	14332	944.0185	13167
watersbp	25.0	19.8	0.3	695.4	2039425	925.5489	1871298
watersym1	71.4	57.1	0.1	13.6	53787	914.5702	48361
watersym2	83.3	55.6	0.1	10.8	28608	1056.1449	25709
waterx	78.6	24.0	0.1	limit	91	–	–
detf1	81.5	1.2	1579.0	5733.5	367	–	–
gear2	70.8	57.6	0.0	0.0	20	0*	13
gear3	50.0	11.1	0.0	0.0	2	0.0164	2
gear4	50.0	22.2	0.0	0.0	4	495720.675	4
gear	50.0	11.1	0.0	0.0	2	0.0164	2
johnall	98.9	9.0	63.2	13.0	18	-224.7302*	16
saa_2	81.5	1.2	1579.0	5733.3	367	–	–
water4	65.1	48.3	0.8	5.5	12624	926.9473	10394
waterz	75.4	58.0	0.2	0.1	63	–	–

Table 9: Computing optimal roundings for MINLP instances, using NLP solution, after cuts

	% Vars Fixed			RENS			
Instance	Int	All	TimeS	Time	Nodes	Solution	Found At
beuster	–	–	0.1	–	–	–	–
cecil_13	37.5	30.8	1.2	775.6	1225350	-115630.852	720438
chp_partload	21.4	1.5	17.6	7146.9	9859	–	–
contvar	–	–	1.9	–	–	–	–
csched1	26.7	20.0	0.1	6864.6	51911752	-30639.353*	510093
csched2a	60.0	52.2	3.6	limit	58208	–	–
eg_all_s	85.7	83.1	682.2	6529.9	1220160	–	–
eg_disc2_s	–	–	798.3	–	–	–	–
eg_disc_s	–	–	546.0	–	–	–	–
eg_int_s	–	–	1011.3	–	–	–	–
eniplac	47.8	42.6	0.2	0.0	28	-130450.77	22
enpro48	82.6	73.4	0.1	3.5	28731	198547.396	28731
enpro48pb	82.6	73.4	0.2	2.3	17748	198547.384	17748
enpro56	68.6	56.8	0.2	8.0	75178	271493.619	75178
enpro56pb	68.6	56.8	0.1	4.7	41949	271496.644	41949
ex1233	0.0	0.0	436.5	7042.5	8215104	–	–
ex1244	0.0	0.0	0.2	0.4	562	82042.2724*	307
ex1252a	0.0	60.0	4.0	0.0	0	–	–
ex1252	28.6	33.9	1.6	1.4	317	131123.771	292
feedtray	14.3	0.4	25.3	limit	406512	–	–
fo7_2	0.0	0.0	0.1	135.9	704358	17.7493*	2293
fo7_ar2_1	0.0	0.0	0.2	46.8	247054	24.8398*	19889
fo7_ar25_1	0.0	0.0	0.1	24.6	115558	23.0936*	105003
fo7_ar3_1	0.0	0.0	0.1	136.1	668929	22.5175*	17122
fo7_ar4_1	0.0	0.0	0.1	155.0	733240	20.7298*	350369
fo7_ar5_1	0.0	0.0	0.1	151.7	767719	17.7493*	68937
fo7	0.0	0.0	0.1	497.1	2372596	20.7298*	240205
fo8_ar2_1	0.0	0.0	0.2	934.6	3788852	30.3406*	1263812
fo8_ar25_1	0.0	0.0	0.2	1106.3	4787074	28.0452*	1555470
fo8_ar3_1	0.0	0.0	0.2	231.3	898814	23.9101*	126001
fo8_ar4_1	0.0	0.0	0.2	234.6	969121	22.3819*	214458
fo8_ar5_1	0.0	0.0	0.1	1432.4	5813287	22.3819*	1898654
fo8	0.0	0.0	6.5	7001.8	26796040	22.3819*	316351
fo9_ar2_1	0.0	0.0	12.1	7024.9	22193275	32.625*	1452885
fo9_ar25_1	0.0	0.0	24.3	7023.6	22803832	32.25	20506093
fo9_ar3_1	0.0	0.0	0.2	1052.7	3352680	24.8155*	336767
fo9_ar4_1	0.0	0.0	16.7	7033.6	28964871	23.4643*	1012573
fo9_ar5_1	0.0	0.0	13.7	7024.4	20112356	23.4643*	1774865
fo9	0.0	0.0	30.3	7040.1	22676841	26.4643	15213281
fuzzy	16.4	6.3	7.8	0.0	3	–	–
gasnet	90.0	39.9	6.3	limit	191339	–	–
ghg_1veh	0.0	0.0	382.9	7085.1	6381143	–	–
ghg_2veh	0.0	0.0	56.0	7146.3	1083873	–	–
ghg_3veh	17.1	21.3	33.2	7164.7	1775681	–	–
hda	14.3	7.1	32.8	7149.4	1682642	–	–
m6	0.0	0.0	0.1	4.1	24562	82.2569*	6680
m7_ar2_1	0.0	0.0	0.2	2.1	10276	190.235*	3930
m7_ar25_1	0.0	0.0	0.2	1.1	3726	143.585*	138
m7_ar3_1	0.0	0.0	0.0	6.3	28008	143.585*	1817
m7_ar4_1	0.0	0.0	0.1	9.3	44016	106.7569*	15850
m7_ar5_1	0.0	0.0	0.0	32.8	173785	106.46*	53909
m7	0.0	0.0	0.1	8.2	48013	106.7569*	20018
mbtd	–	–	limit	–	–	–	–
no7_ar2_1	0.0	0.0	0.2	219.9	1033148	107.8153*	325747
no7_ar25_1	0.0	0.0	0.1	379.1	1545736	107.8153*	548721

Table 9 continued

	% Vars Fixed		RENS				
Instance	Int	All	TimeS	TimeR	NodesR	Solution	Found At
no7_ar3_1	0.0	0.0	0.1	506.6	1988914	107.8153*	118955
no7_ar4_1	0.0	0.0	0.1	2571.1	13791699	98.5184*	9316640
no7_ar5_1	0.0	0.0	0.2	3548.9	14641250	90.6227*	2261480
nvs09	–	–	0.2	–	–	–	–
nvs20	0.0	0.0	0.0	1.2	1668	230.9221*	1585
o7_2	0.0	0.0	42.7	7032.6	26786207	116.9459*	19601790
o7_ar2_1	0.0	0.0	0.2	403.4	1959250	140.4119*	360093
o7_ar25_1	0.0	0.0	0.2	1184.8	4608236	140.7327	293836
o7_ar3_1	0.0	0.0	0.2	2486.6	9747119	137.9318*	3672646
o7_ar4_1	0.0	0.0	4.4	7040.1	26611055	131.6531*	3627436
o7_ar5_1	0.0	0.0	0.9	6992.7	30028960	116.9458*	3480829
o7	0.0	0.0	27.8	7014.7	26516141	131.6531*	544651
o8_ar4_1	0.0	0.0	23.1	7088.4	18402307	245.4744	8887518
o9_ar4_1	0.0	0.0	46.7	7025.3	19840728	250.1082	9730833
oil2	0.0	0.0	35.8	7133.9	1001767	–	–
oil	0.0	0.1	33.7	7149.0	119377	–	–
parallel	20.0	14.7	14.4	limit	900521	924.225	834864
pump	33.3	40.0	0.9	5.7	146	131123.769	143
risk2b	0.0	0.0	0.1	0.1	53	-55.8761*	25
spring	0.0	0.0	0.1	0.0	44	0.9876	34
st_e32	83.3	40.6	0.1	0.0	1	–	–
stockcycle	24.3	21.8	2.6	7159.6	6417875	128864.597	3237213
super1	16.1	1.1	13.5	0.0	1	–	–
super2	16.1	1.2	10.8	0.0	1	–	–
super3	21.6	2.7	16.6	0.0	1	–	–
super3t	0.0	0.0	7.4	7196.9	48370	–	–
synheat	0.0	0.0	3.6	limit	512341	–	–
synthes1	0.0	0.0	0.0	0.0	5	6.0098*	4
synthes2	0.0	0.0	0.0	0.0	16	73.0353*	12
synthes3	0.0	0.0	0.0	11.4	172409	68.0098*	172409
tls12	29.6	67.2	45.9	7125.8	8583097	–	–
tls4	27.1	28.2	0.3	14.0	85190	11.5	4135
tls5	34.4	36.4	0.4	136.8	663149	12.1	49484
tls6	45.5	50.7	0.3	275.6	1106419	–	–
tls7	72.4	78.5	0.5	0.3	965	–	–
water3	3.6	6.3	51.4	6929.7	20651925	908.5771	11154642
waterful2	64.3	58.0	233.9	6956.7	21237400	1727.7383	12114
watersbp	3.6	6.3	139.1	6965.6	21701297	926.9473	1393039
watersym1	42.9	38.0	41.6	6934.4	24032000	945.8494	823376
watersym2	50.0	41.2	0.6	1649.9	5395774	955.728	1697926
waterx	0.0	0.0	7.4	6967.9	983262	–	–
detf1	41.0	0.6	1599.0	5649.0	608	–	–
gear2	0.0	0.0	0.0	0.2	896	-0*	896
gear3	0.0	0.0	0.0	0.0	5	0*	4
gear4	0.0	0.0	0.0	0.0	5	333.1514	4
gear	0.0	0.0	0.0	0.0	5	0*	4
johnall	0.0	0.0	63.6	8.2	1	-224.7302*	1
saa_2	41.0	0.6	1601.4	5649.3	608	–	–
water4	64.3	54.6	0.7	0.8	2430	1008.4471	1819
waterz	65.1	44.4	0.6	36.6	98729	2600.6081	98389

Table 10: Analyzing rounding heuristics for MIPLIB instances

Instance	RENS	ZI Round	Rounding	Simple Rounding
a1c1s1	13209.184	–	–	–
aflow30a	1158	–	–	–
aflow40b	1179	–	–	–
atlanta-ip	98.009586	–	–	–
beasleyC3	789	1690	1730	1730
bell3a	878430.32	880414.28	–	–
biella1	3278480.6	–	–	–
bienst2	54.6	–	–	–
binkar10_1	6746.64	–	–	–
blend2	7.598985	–	–	–
bley_xl1	190	–	–	–
cap6000	-2443599	-2443599	-2441736	-2441736
core2536-691	695	1103	1651	–
cov1075	20	43	90	90
dano3mip	762.75	–	–	–
danoint	65.666667	–	–	–
dcmulti	188186.5	–	–	–
dfn-gwin-UUM	39920	199352	209984	209984
dsbmip	-305.19817	–	–	–
egout	568.1007	597.46403	597.46403	597.46403
fast0507	177	315	540	540
fiber	411151.82	–	–	–
fixnet6	3997	10723.928	10723.928	10723.928
gesa2-o	25780031	–	–	–
gesa2	25780031	–	–	–
gesa3	27991430	–	–	–
gesa3_o	27991430	–	–	–
glass4	2.2666856e+09	–	–	–
gmu-35-40	-2399398.2	–	–	–
gt2	21166	21166	–	–
iis-100-0-cov	29	55	100	100
iis-bupa-cov	36	71	144	144
iis-pima-cov	33	66	130	130
khb05250	1.0694023e+08	1.1688827e+08	1.1688827e+08	1.1688827e+08
liu	3418	–	–	–
lseu	1148	–	–	–
m100n500k4r1	-22	-9	0	0
map18	-847	–	–	–
map20	-922	–	–	–
markshare1	142	584	2108	2108
markshare2	131	531	2288	2288
mas74	14343.468	–	–	–
mas76	40560.054	–	–	–
mcsched	213768	–	–	–
mik-250-1-100-1	-66729	-66409	-66409	-66409
mine-90-10	-7.8430234e+08	–	–	–
misc06	12850.861	12920.927	12920.927	12920.927
mitre	115155	–	–	–
mkc	-539.866	–	–	–
mod008	309	452	1212	1212
mod011	-54219146	–	–	–
modglob	20799459	21051934	21051934	21051934
n3div36	151600	230600	562600	–
n3seq24	68000	–	–	–
n4-3	9010	20686.357	23686.357	23686.357
neos-1337307	-202133	–	–	–

Table 10 continued

Instance	RENS	ZI Round	Rounding	Simple Rounding
neos-1396125	3000.0553	–	–	–
neos13	-65.655161	–	–	–
neos-476283	406.81233	–	–	–
neos-934278	1332	–	–	–
newdano	66.5	–	–	–
ns1758913	-457.71835	–	–	–
nsrand-ipx	55360	–	114560	–
opm2-z7-s2	-10271	-3937	–	–
opt1217	-16	–	–	–
p0201	7805	–	–	–
p0282	258411	400676	373318	–
p0548	8763	–	–	–
p2756	3152	–	–	–
pg5_34	-14287.702	–	–	–
pk1	29	–	–	–
pp08a	7360	12657.971	12657.971	12657.971
pp08aCUTS	7370	13128.015	13128.015	13128.015
qiu	-132.87314	1805.1771	1805.1771	1805.1771
qnet1	21237.655	–	–	–
qnet1_o	22600.83	–	45561.556	–
rail507	178	319	550	–
ran16x16	3846	10305.599	10305.599	10305.599
reblock67	-34629816	–	–	–
rentacar	30356761	–	–	–
rgn	82.199998	–	–	–
rmatr100-p10	424	–	–	–
rmatr100-p5	976	–	–	–
rmine6	-457.17275	-435.70014	–	–
rococoC10-001000	12067	–	87872	–
roll3000	14193	–	–	–
set1ch	54537.75	59480.277	59480.277	59480.277
seymour	427	590	757	757
sp98ic	4.6976602e+08	6.9404931e+08	1.3685495e+09	–
stein27	18	20	27	27
stein45	30	37	45	45
timtab1	827609	–	–	–
tr12-30	131438	–	–	–
vpm2	13.75	–	–	–
zib54-UUE	10334016	19016948	19016948	19016948

Table 11: RENS compared to other primal heuristics, MIPLIB instances

Instance	all heuristics	RENS	Feasibility Pump
a1c1s1	16631.684	–	–
aflow30a	4606	1158	–
aflow40b	8300	–	–
app1-2	-23	–	-23
beasleyC3	945	–	877
bell3a	880414.28	878430.32	912403.02
bell5	8975498.7	–	11608253
biella1	2.794433e+08	3630095.5	3309837.4
bienst2	85.5	–	–
blend2	–	7.598985	–
cap6000	-2451186	-2443599	-2448325
core2536-691	819	701	694
cov1075	27	–	33
dano3mip	847.81818	763.625	–
dcmulti	189453.4	–	189453.4
dfn-gwin-UUM	100020	–	138300
disctom	-5000	–	-5000
ds	5418.56	–	–
dsbmip	-305.19817	–	-305.19817
egout	568.1007	568.1007	610.22138
eil33-2	3376.7853	–	–
eilB101	3109.9773	–	–
fast0507	240	177	198
fiber	514321.26	411151.82	964345.33
fixnet6	4536	3997	4536
flugpl	1322700	–	–
gesa2-o	26755195	25780031	–
gesa2	26443646	25780031	–
gesa3	28239091	27991430	–
gesa3_o	28465633	27991430	–
glass4	–	2.2666856e+09	–
gmu-35-40	-2312990.2	-2399398.2	–
gt2	21166	21166	–
harp2	-44025501	–	–
iis-100-0-cov	35	–	52
iis-bupa-cov	49	–	101
iis-pima-cov	45	34	110
khb05250	1.0875131e+08	1.0694023e+08	1.0921306e+08
liu	4762	–	–
lseu	1252	1148	–
m100n500k4r1	-18	-22	-18
macrophage	608	–	–
map18	-608	-847	–
map20	-702	-918	–
markshare1	204	142	133
markshare2	308	131	217
mas74	13755.892	14343.468	–
mas76	45030.693	40560.054	–
mcsched	267801	–	264722
mik-250-1-100-1	-66409	-66729	-10125
mine-90-10	0	–	–
misc06	12864.57	12850.861	12866.961
mitre	115155	115155	–
mkc	-392.358	–	–
mod008	307	309	363
mod011	0	–	–

Table 11 continued

Instance	all heuristics	RENS	Feasibility Pump
modglob	20786787	–	20762355
momentum3	598721.83	–	–
mspp16	363	–	–
mzzv11	0	–	–
mzzv42z	0	–	–
n3div36	199000	151600	–
n3seq24	133800	75800	–
n4-3	15375	–	14195
neos13	-73.31727	-54.293292	-60.800922
neos18	57	–	21
neos-476283	434.22373	406.81233	–
neos-934278	64298	–	316
newdano	92.5	–	–
noswot	-35	–	–
ns1758913	-387.30071	-457.71835	–
nsrand-ipx	73920	57120	185760
nw04	17526	–	17526
opm2-z7-s2	-2444	–	-1480
opt1217	-15	-16	0
p0201	8735	7805	8185
p0282	281009	258411	–
p0548	35561	8763	–
p2756	3220	3152	–
pg5_34	-10357.263	-14287.702	–
pigeon-10	0	–	–
pk1	79	29	–
pp08a	9540	–	8550
pp08aCUTS	10040	–	8250
qiu	1691.1431	–	-40.870237
qnet1	16430.489	21237.655	29903.897
qnet1_o	18484.148	22600.83	26283.04
rail507	263	185	–
ran16x16	4333	4034	4271
reblock67	0	–	–
rgn	82.199998	82.199998	153.6
rmatr100-p10	725	–	–
rmatr100-p5	1448	976	–
rmine6	-292.59425	-449.05697	–
rococoC10-001000	21783	14338	–
rout	2375.25	–	–
set1ch	55351.5	54537.75	61488.25
seymour	482	443	468
sp98ic	6.7634404e+08	–	–
sp98ir	2.9455711e+08	–	–
stein27	19	–	21
stein45	33	–	39
tanglegram1	34171	–	–
tanglegram2	1577	–	–
tr12-30	151095	–	–
triptim1	22.9021	–	22.9031
vpm2	17.75	13.75	26.5
zib54-UUE	18338824	–	12987543

Table 12: RENS compared to other primal heuristics, MIQCP instances

Instance	all heuristics	RENS	Undercover
10bar2	–	2691.7039	–
25bar	–	1045.1823	–
classical_200_0	-0.071487129	-0.084829963	–
classical_200_1	-0.083770619	-0.097035729	–
classical_20_0	-0.063051702	-0.068648323	–
classical_20_1	-0.067785964	–	–
classical_50_0	-0.078004908	-0.081834178	–
classical_50_1	-0.068876528	-0.073682814	–
clay0305m	81611.329	–	–
du-opt5	45.028201	–	546.27998
du-opt	30.48344	–	632.89142
ex1263	30.1	28.3	30.1
ex1266	27.3	21.3	27.3
fac3	38310066	–	38310066
feedtray2	0	–	–
ibell3a	890253.14	878785.03	915693.16
icvxqp1	526240	914601	526240
imod011	0	–	4.0558269e+08
iportfolio	0	–	–
itointqor	0	53624064	71120986
ivalues	0	9026.4463	23155.091
meanvarx	14.824808	14.369221	14.824808
netmod_dol1	0	–	0
netmod_dol2	0	–	0
netmod_kar1	0	–	0
netmod_kar2	0	–	0
nous1	1.6521101	–	–
nous2	1.3843168	–	–
nvs19	-1097.8	–	0
nvs23	-1124.2	–	484.2
robust_100_0	-0.07383209	-0.088786652	–
robust_100_1	-0.030068722	-0.052515237	–
robust_200_0	-0.083079202	-0.094355298	–
robust_20_0	-0.075867238	-0.075868453	–
robust_50_0	-0.074184525	-0.067067685	–
robust_50_1	-0.05304023	-0.07143226	–
shortfall_100_0	-1.0737261	-1.0737263	–
shortfall_100_1	-1.0459995	-1.0656589	–
shortfall_200_0	-1.0803073	-1.0803069	–
shortfall_20_0	-1.0782714	-1.0810933	–
shortfall_50_0	-1.0799126	-1.0799127	–
shortfall_50_1	-1.0711488	-1.0806174	–
SLay05H	66202.063	24809.675	–
SLay05M	64352.815	33732.861	112668.04
SLay07M	139187.44	73105.885	–
SLay10H	527790	–	–
SLay10M	972591.88	270920.73	–
spectra2	19.284089	13.978303	306.3343
tln5	15.1	–	15.1
tln6	32.3	–	32.3
tln7	30.3	–	30.3
tloss	27.3	–	27.3
tltr	61.133333	–	61.133333
uflquad-15-60	1440.866	–	1440.866
uflquad-20-50	409.43207	–	409.43207
uflquad-40-80	522.98402	–	879.81492

Table 12 continued

Instance	all heuristics	RENS	Undercover
util	1012.1654	1000.9676	1012.18
waste	672.99221	692.78243	672.99221

Table 13: RENS compared to other primal heuristics, MINLP instances

Instance	all heuristics	RENS	Undercover
csched1	-29279.168	-29775.988	–
csched2a	-137442.84	–	–
eg_all_s	16.772411	–	–
eg_int_s	100000	–	–
enpro48	281126.48	241151.11	–
enpro48pb	276126.83	264033.48	–
enpro56	280379.39	289617.34	–
enpro56pb	280379.39	279704.74	–
ex1244	87646.293	–	87646.293
ex1252a	152875.46	–	–
fo7_2	26.12553	–	–
ghg_1veh	7.8438354	–	–
m7_ar5_1	200.46001	–	–
nvs09	-9.7637013	–	28.865663
pump	152875.46	–	–
risk2b	-32.04093	–	-32.04093
stockcycle	306163.25	334280.19	357714.33
synthes1	6.0097585	–	6.0097589
synthes2	83.388996	73.035308	–
synthes3	85.513943	–	–
tls4	–	11.5	–
tls5	–	21.6	–
watersym1	–	950.1639	–
detf1	12.881782	–	–
gear2	1.3353082e-05	0	–
gear3	0.41851773	–	–
gear4	855720.67	–	–
gear	0.41851773	–	–
johnall	-224.73017	–	-224.73016
saa_2	12.881782	–	–
water4	1209.0444	1012.1499	–

Table 14: Impact of RENS on overall solving process for MIPLIB instances

Instance	No RENS		Root RENS		Tree RENS	
	Nodes	Time	Nodes	Time	Nodes	Time
10teams	2 766	33.8	2 766	33.8	2 766	33.8
30n20b8	>13 609	limit	>13 098	limit	>13 480	limit
a1c1s1	>444 580	limit	>445 106	limit	>355 340	limit
acc-tight5	2 414	388.9	2 414	389.5	2 414	389.5
aflow30a	3 617	20.8	1 931	13.2	1 931	13.3
aflow40b	366 800	3221.7	230 705	1087.1	230 705	1085.1
air04	272	77.8	272	77.5	272	77.8
air05	478	45.8	478	44.5	478	44.5
app1-2	76	1139.8	76	1300.6	76	1302.4
arki001	2 703 497	4529.0	2 703 497	4527.6	2 703 497	4526.7
ash608gpia-3col	10	69.7	10	70.0	10	69.9
atlanta-ip	>8 841	limit	>8 520	limit	>8 520	limit
beasleyC3	>1 897 819	limit	>1 890 444	limit	>1 767 779	limit
bab5	>21 663	limit	>21 663	limit	>21 636	limit
bell3a	47 240	13.2	46 910	11.2	46 910	11.1
bell5	1 069	0.6	1 069	0.7	1 069	0.5
biella1	10 546	2284.0	2 607	939.9	2 607	953.5
bienst2	73 759	394.5	73 759	396.7	82 826	454.9
binkar10_1	105 531	158.8	105 531	159.3	129 286	204.9
blend2	2 135	1.9	164	0.7	164	0.9
bley_xl1	18	372.2	1	214.1	1	206.8
bnatt350	7 866	972.6	7 866	970.9	7 866	972.6
cap6000	3 005	2.5	3 005	2.6	3 005	2.8
core2536-691	204	383.3	281	652.9	281	653.5
cov1075	>1 719 951	limit	>1 721 430	limit	>1 697 293	limit
csched010	940 018	6394.7	940 018	6395.9	940 018	6397.6
dano3mip	>2 838	limit	>3 064	limit	>2 384	limit
danoint	1 063 562	5251.8	1 063 562	5237.1	1 063 562	5256.0
dcmulti	130	1.8	130	1.8	130	1.7
dfn-gwin-UUM	77 613	148.8	77 613	146.7	77 613	148.1
disctom	1	3.5	1	3.6	1	3.5
ds	>465	limit	>460	limit	>460	limit
dsbmip	1	0.7	1	0.6	1	0.6
egout	1	0.5	1	0.5	1	0.5
eil33-2	10 571	98.0	10 571	99.3	10 571	99.7
eilB101	9 239	773.3	9 239	777.1	9 239	776.3
enigma	1 289	0.6	1 289	0.6	1 289	0.7
enlight13	1 099 066	655.3	1 099 066	658.3	1 099 066	658.8
enlight14	156 998	108.9	156 998	108.3	156 998	108.1
fast0507	1 477	1474.5	2 774	3501.8	2 774	3509.4
fiber	78	1.9	32	1.3	32	1.2
fixnet6	54	1.8	14	1.8	14	1.9
flugpl	121	0.5	121	0.5	121	0.5
gesa2-o	55	1.8	4	1.5	4	1.5
gesa2	42	1.7	7	1.4	7	1.3
gesa3	147	2.3	16	1.7	16	1.6
gesa3_o	119	3.1	12	2.1	12	2.0
glass4	>10 167 913	limit	1 795 478	1454.2	1 795 478	1459.6
gmu-35-40	>5 151 788	limit	>11 990 260	limit	>13 431 923	limit
gt2	1	0.5	1	0.5	1	0.5
harp2	360 980	301.6	360 980	301.2	364 890	308.2
iis-100-0-cov	106 874	1706.4	106 874	1705.5	106 389	1828.4
iis-bupa-cov	183 185	6723.2	189 467	6655.7	189 467	6690.3
iis-pima-cov	13 766	952.6	13 011	953.7	13 011	966.1
khb05250	11	0.5	11	0.5	11	0.5

Table 14 continued

Instance	No RENS		Root RENS		Tree RENS	
	Nodes	Time	Nodes	Time	Nodes	Time
l152lav	52	3.0	52	2.9	52	3.1
lectsched-4-obj	11 988	246.4	11 988	246.4	11 988	247.4
liu	>1 835 353	limit	>1 832 824	limit	>1 965 400	limit
lseu	329	0.5	552	0.5	552	0.5
m100n500k4r1	5 272 016	4732.9	>8 222 511	limit	>8 183 822	limit
macrophage	>929 901	limit	>925 398	limit	>928 739	limit
map18	607	649.6	293	463.1	293	463.8
map20	1 180	496.4	353	549.0	353	548.5
markshare1	>75 355 137	limit	>78 655 002	limit	>78 886 991	limit
markshare2	>63 825 711	limit	>62 613 242	limit	>62 433 221	limit
mas74	2 955 765	500.1	2 955 765	499.8	2 955 765	502.1
mas76	243 004	43.5	281 857	42.2	281 857	42.3
mcsched	16 113	222.9	16 113	222.2	20 712	256.3
mik-250-1-100-1	1 920 723	373.9	1 021 375	205.6	1 021 375	206.1
mine-90-10	469 802	1753.4	359 569	1156.5	359 569	1157.1
misc03	131	1.1	131	1.2	131	1.1
misc06	18	0.5	6	0.5	6	0.5
misc07	38 363	20.5	38 363	20.5	38 363	20.9
mitre	1	4.5	1	4.6	1	4.7
mkc	>3 288 146	limit	>3 186 952	limit	>3 223 059	limit
mod008	192	0.9	192	0.9	192	0.9
mod010	4	0.9	4	0.7	4	0.8
mod011	1 596	206.1	1 596	206.0	1 596	205.8
modglob	1 408	1.3	1 408	1.5	1 408	1.6
momentum1	>21 781	limit	>21 733	limit	>21 781	limit
momentum2	>63 180	limit	>61 812	limit	>62 495	limit
momentum3	>44	limit	>43	limit	>44	limit
msc98-ip	>756	limit	>756	limit	>756	limit
mspp16	>750	limit	>382	limit	>736	limit
mzzv11	2 734	341.8	2 734	343.5	2 734	342.3
mzzv42z	1 557	364.5	1 557	364.2	1 557	365.0
n3div36	>200 784	limit	>257 302	limit	>264 668	limit
n3seq24	>2 290	limit	>2 094	limit	>2 114	limit
n4-3	53 959	835.6	53 959	835.3	53 959	844.5
neos-1109824	24 162	185.9	24 162	185.4	24 162	186.1
neos-1337307	>415 472	limit	>416 447	limit	>413 169	limit
neos-1396125	54 219	3981.6	54 219	3981.4	54 219	3982.6
neos13	>28 166	limit	>26 778	limit	>25 527	limit
neos-1601936	>31 161	limit	>30 882	limit	>30 831	limit
neos18	9 133	41.4	9 133	41.4	9 133	41.5
neos-476283	466	326.9	609	323.2	609	327.1
neos-686190	9 894	114.1	9 894	114.7	9 894	114.3
neos-849702	137 579	1652.0	137 579	1651.7	137 579	1653.2
neos-916792	57 471	228.0	57 471	227.3	57 471	227.3
neos-934278	>2 951	limit	>4 825	limit	>4 708	limit
net12	3 838	2650.2	3 838	2647.9	3 838	2649.5
netdiversion	>72	limit	>72	limit	>72	limit
newdano	>1 570 960	limit	>1 574 108	limit	>1 138 936	limit
noswot	525 460	148.2	525 460	147.8	525 460	147.4
ns1208400	15 050	1960.2	15 050	1957.1	15 050	1956.6
ns1688347	17 807	1979.0	17 807	1978.5	17 807	1979.6
ns1758913	>23	limit	>17	limit	>5	limit
ns1766074	946 987	514.1	946 987	515.2	946 987	516.1
ns1830653	57 234	584.3	57 234	585.5	57 234	585.9
nsrand-ipx	>1 097 182	limit	>1 154 058	limit	>1 158 945	limit
nw04	5	51.1	5	52.0	5	51.9
opm2-z7-s2	4 401	1154.7	4 401	1153.8	4 401	1154.5
opt1217	>16 012 029	limit	>12 726 890	limit	>12 478 488	limit

Table 14 continued

Instance	No RENS		Root RENS		Tree RENS	
	Nodes	Time	Nodes	Time	Nodes	Time
p0201	169	1.9	65	1.6	65	1.8
p0282	26	0.8	3	0.6	3	0.5
p0548	96	0.8	14	0.5	14	0.5
p2756	403	3.2	153	2.6	153	2.5
pg5_34	348 765	1717.1	318 742	1501.1	306 428	1374.3
pigeon-10	>7 056 792	limit	>7 034 031	limit	>6 972 773	limit
pk1	213 670	46.8	226 780	50.0	206 727	44.4
pp08a	590	1.5	590	1.5	670	1.7
pp08aCUTS	403	1.5	403	1.4	480	1.6
protfold	>6 866	limit	>6 865	limit	>6 862	limit
pw-myciel4	647 355	5306.6	647 355	5310.9	647 355	5311.9
qiu	11 012	56.2	11 012	56.3	10 301	55.9
qnet1	7	2.4	7	2.5	7	2.3
qnet1_o	29	3.9	29	4.0	29	3.9
rail507	1 704	1494.8	1 472	1269.2	1 472	1268.4
ran16x16	348 556	196.6	331 635	195.2	331 635	195.3
reblock67	111 964	279.5	111 964	279.1	111 964	279.7
rd-rplusc-21	>58 623	limit	>58 592	limit	>58 592	limit
rentacar	14	3.0	14	3.0	14	3.1
rgn	62	0.5	62	0.5	62	0.5
rmatr100-p10	901	197.3	901	197.7	864	201.0
rmatr100-p5	420	668.8	385	553.4	385	553.4
rmine6	541 456	2814.6	727 632	4044.6	523 315	2760.6
rocII-4-11	40 353	544.4	40 353	545.6	40 353	545.7
rococoC10-001000	662 755	3313.2	488 147	2372.7	495 582	2404.2
roll3000	>1 390 052	limit	>1 479 602	limit	>1 482 101	limit
rout	29 656	39.7	29 656	39.9	19 937	33.3
satellites1-25	9 089	2148.3	9 089	2146.1	9 089	2148.0
set1ch	28	0.9	6	0.8	6	0.9
seymour	>122 156	limit	>130 095	limit	>116 911	limit
sp98ic	>135 751	limit	>209 889	limit	>208 547	limit
sp98ir	4 912	64.8	4 912	64.9	4 912	65.1
stein27	4 045	0.9	4 045	1.1	4 045	1.0
stein45	52 523	13.1	52 523	13.1	52 523	13.3
swath	>1 448 548	limit	>1 460 957	limit	>1 433 029	limit
t1717	>734	limit	>720	limit	>734	limit
tanglegram1	27	867.6	27	866.3	27	860.5
tanglegram2	3	7.0	3	7.0	3	6.9
timtab1	925 706	412.1	925 706	413.2	925 706	414.5
timtab2	>8 939 001	limit	>8 943 388	limit	>8 926 669	limit
tr12-30	1 518 459	1986.3	1 685 757	2280.3	1 532 831	2052.5
triptim1	30	2002.7	30	1984.3	30	1993.2
unitcal_7	11 624	1173.8	10 569	1137.6	10 569	1138.7
vpm2	945	1.2	143	1.1	143	1.1
vpphard	>5 521	limit	>5 524	limit	>5 525	limit
zib54-UUE	951 366	5701.2	951 366	5708.5	865 298	4910.0
arithm. mean	1 446 078	2461.4	1 442 400	2427.0	1 443 404	2414.3
geom. mean	7 155	220.3	5 870	209.6	5 810	209.4
sh. geom. mean	11 248	377.2	10 390	366.3	10 346	365.8

Table 15: Impact of RENS on overall solving process for MIQCP instances

Instance	No RENS		Root RENS		Tree RENS	
	Nodes	Time	Nodes	Time	Nodes	Time
10bar2	369	2.3	653	2.8	653	2.9
25bar	>7 936	limit	>3 402	limit	>3 402	limit
classical_200_0	>100 675	limit	>109 742	limit	>109 204	limit
classical_200_1	>152 012	limit	>134 651	limit	>131 226	limit
classical_20_0	172	0.7	127	0.9	127	0.9
classical_20_1	866	1.7	897	1.9	897	2.1
classical_50_0	243 420	1068.1	1 260 971	5287.2	940 699	3782.0
classical_50_1	20 929	74.4	29 760	106.3	29 760	107.9
clay0203m	55	0.5	55	0.5	55	0.5
clay0205m	10 494	4.0	10 494	4.1	10 492	4.5
clay0303m	99	0.5	99	0.5	99	0.5
clay0305m	9 361	4.5	9 361	4.5	9 361	4.5
du-opt5	86	0.5	86	0.5	86	0.5
du-opt	322	0.7	322	0.7	322	0.8
ex1263	199	0.7	199	0.8	199	0.8
ex1266	37	0.7	255	1.1	255	1.1
fac3	6	0.5	6	0.5	6	0.5
feedtray2	1	0.5	1	0.5	1	0.5
ibell3a	44 048	12.9	42 066	13.8	42 066	13.8
icvxqp1	>1 897	limit	>1 893	limit	>1 903	limit
ilaser0	169	3.2	169	3.0	169	3.2
imod011	1	319.2	1	319.4	1	319.4
iportfolio	>21 555	limit	>21 527	limit	>21 279	limit
isqp	>1 706 210	limit	>1 706 576	limit	>1 706 619	limit
ivalues	>153 470	limit	>153 572	limit	>153 088	limit
meanvarx	7	0.5	3	0.5	3	0.5
netmod_dol1	62 794	6077.4	62 794	6049.4	62 028	6115.3
netmod_dol2	192	49.6	192	49.7	150	47.8
netmod_kar1	288	5.9	288	5.9	288	5.8
netmod_kar2	288	6.0	288	5.9	288	5.9
nous1	>5 156 737	limit	>5 154 877	limit	>5 149 665	limit
nous2	2 821	2.2	2 821	2.0	2 821	2.2
nuclear14a	>36 917	limit	>36 932	limit	>53 127	limit
nuclear14b	>73 331	limit	>73 976	limit	>73 751	limit
nvs19	105	0.5	105	0.5	105	0.5
nvs23	96	0.5	96	0.5	96	0.5
product2	>6 014 234	limit	>6 225 476	limit	>5 740 865	limit
product	5 562	11.7	7 747	15.7	7 853	15.9
robust_100_0	86 362	1307.3	79 523	1234.3	79 523	1245.8
robust_100_1	13 780	207.9	16 517	235.9	16 517	239.9
robust_200_0	>139 784	limit	>74 872	limit	>73 339	limit
robust_20_0	8	0.5	8	0.5	8	0.5
robust_50_0	91	1.4	91	1.8	91	1.8
robust_50_1	228	3.0	200	2.8	200	2.8
shortfall_100_0	>495 750	limit	>497 757	limit	>503 010	limit
shortfall_100_1	356 687	3926.5	311 239	3382.3	226 505	2414.0
shortfall_200_0	>104 110	limit	>103 692	limit	>103 523	limit
shortfall_20_0	102	0.8	120	0.9	120	0.8
shortfall_50_0	343 829	1738.6	695 205	3628.8	690 262	3615.6
shortfall_50_1	9 259	43.2	11 106	46.0	11 106	47.4
SLay05H	254	2.1	75	1.6	75	1.6
SLay05M	79	0.6	150	1.0	150	1.0
SLay07M	1 930	6.9	377	3.0	377	3.1
SLay10H	>532 368	limit	>532 759	limit	>498 710	limit
SLay10M	229 809	1828.4	28 848	233.2	28 856	241.4

Table 15 continued

Instance	No RENS Nodes	No RENS Time	Root RENS Nodes	Root RENS Time	Tree RENS Nodes	Tree RENS Time
space25a	>21 026	limit	>21 026	limit	>21 026	limit
space25	>8 751	limit	>8 751	limit	>8 751	limit
spectra2	33	0.7	23	0.7	23	0.8
tln12	>2 590 652	limit	>2 587 580	limit	>2 589 049	limit
tln5	44 527	26.2	44 527	26.1	44 527	26.3
tln6	>12 370 474	limit	>12 372 692	limit	>12 367 087	limit
tln7	>9 474 819	limit	>9 482 513	limit	>9 493 095	limit
tloss	60	0.5	60	0.5	60	0.5
tltr	24	0.5	24	0.5	24	0.5
uflquad-15-60	904	2857.7	904	2862.1	827	2491.9
uflquad-20-50	>201	limit	>201	limit	>34	limit
uflquad-40-80	>105	limit	>105	limit	>39	limit
util	371	0.5	375	0.5	375	0.5
waste	>4 005 594	limit	>3 983 731	limit	>3 964 173	limit
arithm. mean	659 740	2872.3	677 123	2927.0	664 117	2888.6
geom. mean	3 823	84.5	3 742	86.4	3 561	86.2
sh. geom. mean	6 457	229.9	6 361	232.0	6 193	229.9

Table 16: Impact of RENS on overall solving process for MINLP instances

Instance	No RENS Nodes	No RENS Time	Root RENS Nodes	Root RENS Time	Tree RENS Nodes	Tree RENS Time
beuster	>243	limit	>243	limit	>243	limit
cecil_13	>2 557 284	limit	>2 553 413	limit	>2 568 736	limit
contvar	>10 024	limit	>10 024	limit	>10 024	limit
csched1	44 649	17.2	44 649	17.5	44 649	17.6
csched2a	>26 250	limit	>26 250	limit	>26 250	limit
detf1	>331	limit	>330	limit	>331	limit
eg_all_s	>446	limit	>440	limit	>440	limit
eg_disc2_s	>83	limit	>83	limit	>48	limit
eg_disc_s	>136	limit	>136	limit	>34	limit
eg_int_s	>5	limit	>5	limit	>5	limit
eniplac	172	0.7	172	0.6	98	0.6
enpro48	84	0.8	54 982	11.9	12 571	4.3
enpro48pb	249 160	42.9	36	0.9	36	0.8
enpro56pb	4 048	1.8	85 265	17.6	85 265	17.6
ex1233	>11 127 294	limit	>11 141 457	limit	>11 144 945	limit
ex1244	492	1.0	492	1.1	504	1.4
ex1252	>88	limit	>88	limit	>88	limit
ex1252a	>204	limit	>204	limit	>204	limit
feedtray	>640 421	limit	>638 931	limit	>639 220	limit
fo7	163 542	68.1	163 542	67.8	163 542	68.6
fo7_2	45 627	22.2	45 627	22.2	48 697	23.8
fo7_ar25_1	43 715	16.9	43 715	17.3	49 960	19.5
fo7_ar2_1	39 986	17.3	39 986	17.3	39 986	17.6
fo7_ar3_1	47 741	17.9	47 741	17.9	50 563	19.5
fo7_ar4_1	58 884	28.5	58 884	28.2	58 884	29.2
fo7_ar5_1	20 509	9.1	20 509	9.0	20 509	9.1
fo8	538 828	277.3	538 828	277.3	538 828	279.3
fo8_ar25_1	337 708	141.8	337 708	141.4	149 658	59.9
fo8_ar2_1	643 114	168.7	643 114	168.6	192 277	75.0
fo8_ar3_1	75 943	43.8	75 943	43.8	75 943	44.6
fo8_ar4_1	>46 231 801	limit	>46 093 488	limit	86 646	43.3
fo8_ar5_1	55 953	27.9	55 953	28.4	55 953	29.2
fo9	2 155 434	1140.4	2 155 434	1143.8	10 127 873	2879.5
fo9_ar25_1	4 702 715	1731.2	4 702 715	1733.8	4 881 081	1843.4
fo9_ar2_1	2 615 019	1089.9	2 615 019	1092.5	2 615 019	1092.2
fo9_ar3_1	532 025	284.5	532 025	284.9	331 077	172.6
fo9_ar4_1	284 985	133.1	284 985	134.6	284 985	133.7
fo9_ar5_1	729 300	405.2	729 300	408.4	729 300	409.3
fuzzy	>2 161 178	limit	>2 156 389	limit	408 344	1883.2
gasnet	>1 382	limit	>1 382	limit	>1 382	limit
gear	2 828	2.0	2 828	2.0	2 828	2.0
gear2	591	0.5	506	0.5	506	0.5
gear3	2 828	2.2	2 828	2.1	2 828	2.0
gear4	105	0.5	105	0.5	105	0.5
ghg_1veh	>18 013 454	limit	>18 137 988	limit	>18 188 182	limit
ghg_2veh	>737 048	limit	>87 992	limit	>853 625	limit
ghg_3veh	>420 745	limit	>420 693	limit	>211 106	limit
hda	>848 500	limit	>847 241	limit	>824 623	limit
johnall	1	64.0	1	72.3	1	63.8
m6	955	1.1	955	1.0	955	1.2
m7	14 053	6.5	14 053	6.4	14 053	6.6
m7_ar25_1	2 848	2.0	2 848	2.1	2 055	1.4
m7_ar2_1	22 707	5.7	22 707	5.6	22 707	5.8
m7_ar3_1	9 390	4.6	9 390	4.5	9 390	4.6
m7_ar4_1	2 134	1.8	2 134	1.8	2 134	2.1

Table 16 continued

Instance	No RENS		Root RENS		Tree RENS	
	Nodes	Time	Nodes	Time	Nodes	Time
m7_ar5_1	25 814	6.8	25 814	6.9	25 814	7.2
no7_ar25_1	107 048	51.4	107 048	50.7	87 297	42.4
no7_ar2_1	27 667	14.9	27 667	14.8	27 667	14.9
no7_ar3_1	423 874	187.2	423 874	185.8	423 874	186.9
no7_ar4_1	228 710	108.6	228 710	108.3	252 173	120.5
no7_ar5_1	103 053	52.5	103 053	52.2	103 053	52.0
nvs09	>4 697 821	limit	>6 241 826	limit	>6 342 072	limit
nvs20	355	0.8	355	0.8	355	1.0
o7	4 566 673	2343.0	4 566 673	2345.9	4 566 673	2357.2
o7_2	1 730 061	756.5	1 730 061	754.7	1 708 453	755.9
o7_ar25_1	489 625	241.3	489 625	239.7	489 625	244.1
o7_ar2_1	176 585	88.0	176 585	86.2	151 581	69.9
o7_ar3_1	1 230 419	616.6	1 230 419	616.7	1 230 419	618.9
o7_ar4_1	1 854 132	991.8	1 854 132	994.0	1 854 132	994.5
o7_ar5_1	795 136	371.7	795 136	372.3	613 092	282.3
o8_ar4_1	11 782 816	6666.4	11 782 816	6688.3	12 722 339	6984.3
o9_ar4_1	>12 507 230	limit	>12 514 424	limit	>12 415 746	limit
oil	>589 974	limit	>589 231	limit	>589 208	limit
oil2	>1 027 176	limit	>1 028 096	limit	>1 024 608	limit
parallel	735 814	2599.6	735 814	2592.5	735 814	2591.3
pump	>47	limit	>47	limit	>47	limit
risk2b	2	0.6	2	0.6	2	0.6
saa_2	>331	limit	>331	limit	>331	limit
spring	90	0.5	90	0.5	90	0.5
st_e32	12 153	13.6	12 153	13.7	12 153	13.6
stockcycle	32 340	222.0	32 340	222.2	32 340	223.2
super1	>88 353	limit	>88 400	limit	>88 430	limit
super2	>90 554	limit	>89 681	limit	>90 164	limit
super3	>102 297	limit	>100 310	limit	>102 024	limit
super3t	>71 449	limit	>71 272	limit	>68 820	limit
synheat	>68 710	limit	>68 710	limit	>68 710	limit
synthes1	4	0.5	4	0.5	4	0.5
synthes2	5	0.5	4	0.5	4	0.5
synthes3	>56 469 781	limit	>54 499 711	limit	>57 219 056	limit
tls12	>622 812	limit	>629 179	limit	>628 973	limit
tls4	9 520	11.7	12 723	13.4	12 723	13.5
tls5	>3 950 998	limit	>3 941 413	limit	>3 943 467	limit
tls6	>2 741 985	limit	>2 729 799	limit	>2 732 632	limit
tls7	>1 805 765	limit	>1 797 325	limit	>1 804 162	limit
water3	>6 706 261	limit	>6 698 169	limit	>6 578 939	limit
water4	1 692 444	1860.5	1 692 444	1863.9	1 642 038	1816.3
waterful2	>4 169 416	limit	>4 164 237	limit	>4 148 024	limit
watersbp	>4 032 620	limit	>4 032 620	limit	>155 142	limit
watersym1	>6 705 227	limit	>6 453 837	limit	>6 730 378	limit
watersym2	>8 127 217	limit	>8 123 253	limit	>8 059 966	limit
waterx	>1 425	limit	>1 425	limit	>1 425	limit
waterz	>1 094 883	limit	>1 094 883	limit	>1 094 883	limit
arithm. mean	2 338 903	3274.5	2 324 208	3274.7	1 925 902	3168.7
geom. mean	45 334	288.0	44 723	291.4	38 568	267.9
sh. geom. mean	58 758	466.5	58 406	467.1	51 066	431.3