Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

TIMO BERTHOLD

RENS– Relaxation Enforced Neighborhood Search

http://www.zib.de/Optimization/Projects/MIP/

ZIB-Report 07-28 (Dezember 2007)

Rens– The Relaxation Enforced Neighborhood Search

Timo Berthold*

April 25, 2008

Abstract

In the recent years, a couple of quite successful large neighborhood search improvement heuristics for MIPs has been published. We present a new start heuristic called Rens for general MIPs working in the spirit of large neighborhood search. It constructs a sub-MIP which represents the space of all feasible roundings of some fractional point – normally the optimum of the LP-relaxation of the original MIP. Thereby, one is able to determine whether a point can be rounded to a feasible solution and which is the best possible rounding. This can be used for the analysis of MIP rounding heuristics. Furthermore, a slightly modified version of Rens proves to be a well-performing heuristic inside the branch-cut-and-price framework SCIP.

Keywords: mixed integer programming, primal heuristics, large neighborhood search

1 Introduction

Solving Mixed Integer Programs (MIPs) is one of the most important techniques to cope with issues that arise in various areas of Combinatorial Optimization and Operations Research. Although MIP-solving is an $\mathcal{NP}$-hard optimization problem, many practically relevant instances can be solved in reasonable time. A MIP is defined by a set of variables, a set of linear constraints, a set of linear constraints and a linear objective function which has to be optimized.

More formally stated:

Definition 1.1. *Let* $\hat{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$. *Let* $m, n \in \mathbb{R}$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $l, u \in \hat{\mathbb{R}}^n$, *and* $I \subseteq N = \{1, \dots, n\}$. *The optimization problem*

$$\begin{array}{lll} \min & c^T x & \\ \textit{such that} & Ax \leq b & \\ & l \leq x \leq u & \\ & x_j \in \mathbb{Z} & \textit{for all } j \in I \end{array} \tag{1}$$

is called a mixed integer program (MIP). *The optimization problem which arises if the integrality constraints are left out, is called the LP (linear program) relaxation of the MIP.*

*Konrad-Zuse-Zentrum für Informationstechnik Berlin, berthold@zib.de

Let $B := \{j \in I \mid l_j = 0, u_j = 1\}$*. We call* $\{x_j \mid j \in I\}$ *the set of* integer variables*,* $\{x_j \mid j \in B\}$ *the set of* binary variables*,* $\{x_j \mid j \in I \setminus B\}$ *the set of* general integer variables*,* $\{x_j \mid j \in N \setminus I\}$ *the set of* continuous variables*.*

The standard approach to solve MIPs are LP-based branch-and-cut algorithms. These are implicit enumeration strategies, for which the solution space is recursively divided into smaller subproblems, building up a so-called branch-and-bound tree. At each node the LP-relaxation of the current subproblem is solved and usually a branching is performed by adding complementary bound changes on an integer variable with a fractional value in the current LP solution.

In state-of-the-art MIP-solvers like Cplex [25] or SCIP [4], primal heuristics play a major role in finding feasible solutions in early steps of the branch-and-bound process (often already at the root node). The knowledge of feasible solutions guides the remaining search process, thereby reduces the overall computational effort, and nevertheless proves the feasibility of the MIP model. Furthermore, a good heuristic solution may be sufficient for practical applications.

Various methods for heuristic MIP-solving have been presented in the literature, including Hillier [24], Balas and Martin [8], Saltzman and Hillier [31], Glover and Laguna [19, 20, 21], Glover et al. [28, 22], Balas et al. [7, 9], Løkketangen [27], Fischetti et al. [17, 16], Danna et al. [14] Bertacco et al. [10], Achterberg and Berthold [2], Eckstein and Nediak [15], Ghosh [18] and Berthold [11].

Broadly speaking, heuristic rounding procedures and Large Neighborhood Search (LNS) strategies are complementary with respect to the computational effort. The running time of rounding heuristics often is linear in the number of fractional variables, whereas LNS heuristics usually have to solve NP-hard subproblems.

In this paper, we will present a new LNS heuristic called Rens, which in contrast to the ones presented in the literature of the recent years does not need an incumbent MIP solution as a start point . On the one hand, it can be used as a tool to evaluate the performance of heuristic rounding methods, on the other hand it proves to be a reasonable start heuristic on its own.

The rest of this article is organized as follows: In the remainder of Section 1 we give a brief introduction into the idea of Large Neighborhood Search and review LNS strategies which have been recently proposed. In Section 2 we introduce our new heuristic and discuss implementational issues. Finally, Section 3 presents some computational results.

Large Neighborhood Search

The *Local Search* [21, 33] approach generalizes the idea of k-OPT [26]: one defines a neighborhood of some reference point, determines a point of this neighborhood which is optimal for some function (e.g., the objective function of the MIP or some feasibility measure), which is then used as a new reference point in the next iteration step. Most improvement heuristics can be formulated as Local Search methods.

Classical Local Search uses relatively small neighborhoods which can be quickly explored and performs a couple of iteration steps, building up a network of visited points. During the recent years, another approach got into the focus of attention.

Large Neighborhood Search or shortly *LNS* is a variant of Local Search. It incorporates the complexity of general MIPs by defining a relatively large neighborhood of some reference point – normally the incumbent solution – and performing just a single iteration step, where the neighborhood is completely or partially searched. Several LNS heuristics [17, 14, 30, 18] have been presented in the literature of the last five years. They all have in common that they use the incumbent solution as starting points and define the neighborhood to be a sub-MIP of the original MIP which is constructed by fixing variables and adding constraints.

The main difference is the way in which the sub-MIP is defined. Two of the proposed methods form a sub-MIP by fixing some of the integer variables, one adds further constraints, and one is a mixture of these approaches.

RINS (Relaxation Induced Neighborhood Search) which was described by Danna, Rothberg, and Le Pape [14] fixes variables which take identic variables in the incumbent and the optimum of the LP-relaxation in the current branch-and-bound node. Crossover which was independently developed by Rothberg [30] and Berthold [12] fixes variables which take identic values in a couple of feasible solutions. Local Branching which was introduced by Fischetti and Lodi [17] adds an additional distance constraint which guarantees that the solutions of the sub-MIP does not differ in more than say k variables from the reference point. DINS as suggested by Ghosh [18] is a mixture of all these strategies.

All of them are improvement heuristics, hence they rely on information of at least one feasible solution. We propose a LNS heuristic, which only uses the optimum of the LP-relaxation of a MIP and can therefore be applied as a start heuristic.

2 Rens

In this section, we will describe an LNS heuristic which investigates the set of all possible roundings of a (fractional) solution of the LP-relaxation.

In many practical applications, a couple of integer variables already takes an integral value in the optimum of the LP-relaxation. The idea is to fix these variables and perform a LNS on the remaining variables. For many MIP-solving techniques binary variables are preferred to general integers. A bound change automatically fixes a binary variable which is useful for branching, and specific algorithm like probing [32], knapsack cover cuts [6, 23, 34], or OCTANE [7] are only used for binary variables. Therefore, we do not only fix variables, but rebound all general integer variables with a fractional LP-value to the nearest integers. Summarized, the sub-MIP is created by changing the bounds of all integer variables x_j to $l_j = \lfloor \bar{x}_j \rfloor$ and $u_j = \lceil \bar{x}_j \rceil$, where $\bar{x}$ is the optimum of the LP-relaxation.

As the overall performance of the heuristic strongly depends on $\bar{x}$ we named it *relaxation enforced neighborhood search*, or shortly RENS.

RENS is of special interest for the analysis of rounding heuristics. If the sub-MIP created by RENS is proven to be infeasible, no rounding heuristic exists which is able to generate a feasible solution out of the fractional LP optimum. If the sub-MIP created by RENS is completely solved, its optimal solutions are the best roundings any pure rounding heuristic can generate.

Implementation Details

For the practical use as a start heuristic integrated into a branch-and-bound framework like SCIP, one should only call RENS, if the resulting sub-MIP seems to be substantially easier than the original one. This means that at least a specific ratio of all integer variables, say r_1, or a specific ratio of all variables including the continuous ones, say r_2, should be fixed.

The first criterion keeps control of the difficulty of the sub-MIP itself, the second one of the LPs that will have to be solved during the solving process. For example, think of a MIP which consists of 20 integer and 10000 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.

Another way to avoid spending too much time in solving sub-MIPs is to add some limit to the solving process of the sub-MIP. This could be a time limit or a limit on the solving nodes.

We decided to limit the number of solving nodes and the number of stalling nodes of the sub-MIP. The solving node limit l_1 is a hard limit on the maximum number of branch-and-bound nodes the MIP-solver should at most process. The stalling node limit l_2 indicates how many nodes the MIP-solver should at most process without an improvement in the incumbent solution of the sub-MIP.

The solving node limit keeps control of the overall running time of the heuristic. On the other hand one does not want to abort the sub-MIP solving too early if the objective value of the incumbent solution keeps increasing during the search process and hence use a stalling node limit. Therefore, we decided to use both node limitation strategies simultaneously with a relatively small stalling node limit l_2 and a large solving node limit l_1.

3 Computational Results

We integrated an implementation of RENS into the branch-and-cut framework SCIP [1, 3, 4]. All computations were made on a 2.20 GHz AMD Opteron with 1024 KB cache and 32 GB RAM.

3.1 Test Set and Settings

We use a wide test set which consists of very different classes of MIP instances. Altogether, there are 129 instances which were taken from:

- the MIPLIB 3.0 [13],
- the MIPLIB 2003 [5], and
- the MIP collection of Mittelmann [29].

Our test set contains all instances of these three collections except for the following: `gen`, `mod010`, `p0033`, `vpm1`, `manna81`, `neos4`, `neos8`, for which the optimum of the LP-relaxation using SCIP default settings is already integer feasible, `momentum3`, `stp3d`, whose root node LPs could not be solved by SCIP within a time limit of half an hour, and `markshare1_1`, `harp2`, which caused numerical troubles when running SCIP with default settings.