Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

TIMO BERTHOLD*, STEFAN HEINZ*,
MARC E. PFETSCH[1], STEFAN VIGERSKE[2,*]

Large Neighborhood Search beyond MIP

[1] Technische Universität Braunschweig, Institut für Mathematische Optimierung, Pockelsstraße 14, 38106 Braunschweig, Germany, m.pfetsch@tu-bs.de
[2] Humboldt-Universität zu Berlin, Institut für Mathematik, Unter den Linden 6, 10099 Berlin, Germany, stefan@math.hu-berlin.de

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

ZIB-Report 11-21 (May 2011)

Large Neighborhood Search beyond MIP

Timo Berthold[1], Stefan Heinz[1], Marc E. Pfetsch[2], Stefan Vigerske[3]

[1] Zuse Institute Berlin
Takustr. 7, 14195 Berlin, Germany
{berthold,heinz}@zib.de

[2] Technische Universität Braunschweig, Institut für Mathematische Optimierung
Pockelsstraße 14, 38106 Braunschweig, Germany
m.pfetsch@tu-bs.de

[3] Humboldt-Universität zu Berlin, Institut für Mathematik
Unter den Linden 6, 10099 Berlin, Germany
stefan@math.hu-berlin.de

Abstract

Large neighborhood search (LNS) heuristics are an important component of modern branch-and-cut algorithms for solving mixed-integer linear programs (MIPs). Most of these LNS heuristics use the LP relaxation as the basis for their search, which is a reasonable choice in case of MIPs. However, for more general problem classes, the LP relaxation alone may not contain enough information about the original problem to find feasible solutions with these heuristics, e.g., if the problem is nonlinear or not all constraints are present in the current relaxation.

In this paper, we discuss a generic way to extend LNS heuristics that have been developed for MIP to constraint integer programming (CIP), which is a generalization of MIP in the direction of constraint programming (CP). We present computational results of LNS heuristics for three problem classes: mixed-integer quadratically constrained programs, nonlinear pseudo-Boolean optimization instances, and resource-constrained project scheduling problems. Therefore, we have implemented extended versions of the following LNS heuristics in the constraint integer programming framework SCIP: LOCAL BRANCHING, RINS, RENS, CROSSOVER, and DINS. Our results indicate that a generic generalization of LNS heuristics to CIP considerably improves the success rate of these heuristics.

1 Introduction

Large neighborhood search (LNS) is a variant of the local search paradigm that has been widely used in Constraint Programming, Operations Research, and Combinatorial Optimization [1, 22, 23]. LNS has proved to be an extremely successful metaheuristic for a wide range of applications in recent years, see, for example, Pisinger and Røpke [24]. The main idea is to restrict the search for "good" solutions to a neighborhood of specific points – usually close to optimal/feasible solutions. The hope is that such a restriction makes the subproblem much easier to solve, while still providing solutions of high quality.

In *mixed-integer linear programming* (MIP), LNS has recently been realized in a series of primal heuristics [5, 6, 13–15, 26]. The so-called LOCAL BRANCHING heuristic has been further extended to constraint programs [16] and mixed-integer nonlinear programs [19].

The goal of this paper is to show that the above mentioned LNS heuristics, which have specifically been developed for MIP, can be extended in a straightforward manner to the broad class of so-called constraint integer programs. We show – via computational experiments for three general problem classes – that this leads to very powerful heuristics. As prototype applications, we consider *mixed-integer quadratically constrained programs* (MIQCPs), *pseudo-Boolean optimization* (PBO), and *resource-constrained project scheduling problems* (RCPSPs). Each of these problems forms a subclass of *constraint integer programs (CIPs)* [3]. CIP adopts modeling and solving techniques from *constraint programming* (CP), MIP, and *satisfiability testing* (SAT). The aim is to restrict the generality of CP modeling as little as needed, while still retaining the full performance of MIP solving techniques.

The contribution of this paper is to investigate the performance of *generic* implementations of LNS heuristics in a complete CIP solver; note that this is in general not competitive with handcrafted problem

specific heuristics. It turns out, that this allows for considerable improvements in finding "good" solutions in the beginning of the solving process for instances in all three considered problem classes. Since the restriction to a neighborhood of some solution again generates a CIP, LNS heuristics are conceptually well suited as generic heuristics for CIPs. Thus, once a CIP solver is able to handle a certain problem class, many LNS heuristics that work on this problem class come at almost no additional implementation cost. We have realized these generic LNS heuristics and the concrete problem classes in the CIP solver SCIP [3, 27]. Its plugin oriented design allows for an easy implementation.

This paper is structured as follows. We first review LNS heuristics in MIP and describe the considered LNS heuristics and problem classes. We then briefly discuss their realization in SCIP, which is a competitive solver for MIQCP [7], PBO [9], and RCPSP [8]. Finally, we present computational results that illustrate the effect of the considered LNS heuristics.

2 Large Neighborhood Search for MIP

A *mixed-integer linear program* is an $\mathcal{NP}$-hard optimization problem, which can be written in the following form

$$\begin{aligned} \min \quad & \boldsymbol{d}^T \boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} \leq \boldsymbol{b}, \\ & x_j \in \mathbb{Z}, \qquad\qquad j \in J, \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$, $\boldsymbol{d} \in \mathbb{R}^n$, $\boldsymbol{b} \in \mathbb{R}^m$, and $J \subseteq \{1, \ldots, n\}$ denotes the subset of integral variables.

Many MIP primal heuristics published in recent years [5, 6, 13–15, 26] are based on large neighborhood search. These heuristics investigate a neighborhood of a small set of starting points such as the best known integral solution (*incumbent*) or the optimal solution of the *linear programming* (LP) relaxation in which the integrality restriction has been dropped. The heuristics create a sub-MIP of the original MIP, typically by fixing some variables to values that are taken from the given points. For problems with binary variables only, another possibility is to add linear constraints, which restrict the number of variables that are different from the given point. By the use of auxiliary variables, this can be extended to problems with general integer variables while maintaining linearity.

Obviously, a good definition of the neighborhood is the crucial point: The neighborhood should contain high quality solutions, these solutions should be easy to find, and the neighborhood should be easy to process. Naturally, these three goals are conflicting in practice. In the remainder of this section, we will give a brief introduction to LNS heuristics for MIPs that have been proposed in the literature of the last ten years.

LOCAL BRANCHING [14] measures the distance to the starting point in Manhattan norm on the integer variables and only considers solutions which are inside a k-neighborhood of the reference solution, where k is typically between 10 and 20. This is done by adding a linear constraint that sums up the distance to the incumbent over all variables. In case of general integer variables, auxiliary variables might have to be used to model the absolute value. For details, see [14].

A "good" MIP solution fulfills three conditions: it is integral, feasible for the linear constraints, and it has a small objective function value (in case of minimization problem). *The relaxation induced neighborhood search* (RINS) [13] uses two starting points: The incumbent MIP solution which fulfills the first two requirements and the optimum of the LP relaxation which fulfills the latter two. RINS defines the neighborhood by fixing all integer variables which take the same value in both solutions.

In contrast to RINS, the *relaxation enforced neighborhood search* (RENS) [6] does not require an incumbent solution. Thus, it can be used as a start heuristic. RENS fixes all integer variables that take an integral value in the optimal solution of the LP relaxation. For the remaining integer variables, the bounds get tightened to the two nearest integral values.

CROSSOVER is an improvement heuristic that is inspired by genetic algorithms [5, 26] and requires more than one feasible solution. For a set of feasible solutions, e.g., the three best found so far, it fixes variables that take identical values in all of them.

RINS, RENS, and CROSSOVER solely fix variables; no further constraints are added to the subproblem.

DINS [15] combines the ideas of RINS and LOCAL BRANCHING. It defines the neighborhood by introducing a distance function between the incumbent solution and the optimum of the LP relaxation. When applied during a branch-and-bound search, it further takes into account how variables change their values at different nodes of the tree.

All those primal heuristics have been developed for mixed-integer *linear* programs. Except for LOCAL BRANCHING [16, 19], the authors are not aware of published extensions to more general problem classes. The mentioned heuristics depend on the existence of a relaxation and/or incumbent solution.

One possibility to extend them to more general problem classes is to apply them to the MIP which results from taking a linear relaxation of the nonlinear problem plus the integrality constraints. In this paper, we show that a more promising way is to create a subproblem by taking a copy of the original problem plus additional constraints specific to the particular LNS heuristic.

3 Constraint Integer Programs

A constraint integer program (CIP) is an optimization problem with a finite number of variables, where a linear function is minimized with respect to a finite set of constraints and with integrality restrictions imposed on all or a subset of the variables. Each constraint is specified by a mapping that indicates whether a given assignment to the variables satisfies this constraint or not. Further, it is required that the subproblem remaining after fixing all integer variables can be solved efficiently. Typically, this subproblem is a linear program [3], but also nonlinear problems are possible, see, e.g., [10]. Note that general objective functions can be modeled by introducing an auxiliary variable that is linked to the actual objective function via an additional constraint.

In this section we introduce the three problem classes which we use for our experiments. These are mixed-integer quadratically constrained programs, pseudo-Boolean optimization, and resource-constrained project scheduling problems.

3.1 Mixed-Integer Quadratically Constrained Programs

A *mixed-integer quadratically constrained program* (MIQCP) is a special case of a CIP in which all constraints are given by either linear or quadratic functions. MIQCPs arise in many areas, for example in mine production scheduling [11]. Formally, an MIQCP can be written in the following form:

$$\begin{array}{lll} \min & \boldsymbol{d}^T \boldsymbol{x} & \\ \text{s.t.} & \boldsymbol{x}^T A_i \, \boldsymbol{x} + \boldsymbol{b}_i^T \, \boldsymbol{x} + c_i \leq 0, & i = 1, \ldots, m \\ & x_j \in \mathbb{Z}, & j \in J, \end{array}$$

where $A_i \in \mathbb{R}^{n \times n}$, $\boldsymbol{b}_i \in \mathbb{R}^n$, and $c_i \in \mathbb{R}$ for $i = 1, \ldots, m$, and $J \subseteq \{1, \ldots, n\}$ denotes the subset of integral variables. If $A_i = 0$, then constraint i is *linear*, otherwise *quadratic*. If A_i is positive semidefinite, then the constraint i is called *convex*, otherwise *nonconvex*. Note that quadratic objective functions can be handled by including an appropriate constraint and an artificial variable.

In the presence of nonconvex constraints, SCIP applies a spatial branch-and-bound approach [10], i.e., branching on continuous variables. As a consequence, the sizes of nonconvex problems that can be solved efficiently are by several orders of magnitude smaller than those of convex problems. SCIP is a competitive solver for MIQCP, see [10, 17].

Test Set. For our experiments we used the test set introduced in [7], from which we removed instances that SCIP reformulates as MIPs during presolve. This leads to a test set of 64 instances.