European Journal of Operational Research 263 (2017) 62–71

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A hybrid primal heuristic for finding feasible solutions to mixed integer programs

Carlos E. Andrade[a,*], Shabbir Ahmed[a], George L. Nemhauser[a], Yufen Shao[b]

[a] H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, 755 Ferst Drive NW, Atlanta, GA 30332, USA
[b] ExxonMobil Upstream Research Company, 3120 Buffalo Speedway, Houston, TX 77098, USA

ARTICLE INFO

Article history:
Received 17 June 2016
Accepted 3 May 2017
Available online 9 May 2017

Keywords:
Integer programming
Primal heuristics
Feasibility

ABSTRACT

We present a new framework for finding feasible solutions to mixed integer programs (MIP). We use the feasibility pump heuristic coupled to a biased random-key genetic algorithm (BRKGA). The feasibility pump heuristic attempts to find a feasible solution to a MIP by first rounding a solution to the linear programming (LP) relaxation to an integer (but not necessarily feasible) solution and then projecting it to a feasible solution to the LP relaxation. The BRKGA is able to build a pool of projected and rounded but not necessarily feasible solutions and to combine them using information from previous projections. This information is also used to fix variables during the process to speed up the solution of the LP relaxations, and to reduce the problem size in enumeration phases. Experimental results show that this approach is able to find feasible solutions for instances where the original feasibility pump or a commercial mixed integer programming solver often fail.

1. Introduction

An important feature of modern mixed integer programming (MIP) algorithms is the ability to effectively find feasible solutions quickly. A variety of heuristic methods have been developed for this purpose. See (Berthold & Hendel, 2015; Fischetti & Salvagnin, 2009; Glover & Laguna, 1997a, 1997b) for surveys of these heuristics. The *Feasibility Pump* heuristic from Fischetti, Glover, and Lodi (2005) has received a lot of attention because of its ability and efficiency to find feasible solutions relatively quickly. Several commercial and open source solvers use internal implementations of the feasibility pump. The feasibility pump and its improvements and variants (Achterberg & Berthold, 2007; Baena & Castro, 2011; Boland et al., 2014; Fischetti & Salvagnin, 2009; Santis, Lucidi, & Rinaldi, 2014) are built on a clever and straightforward idea for finding a feasible solution to a MIP by first rounding a solution to the LP relaxation to an integer (but not necessarily feasible) solution and then projecting it to a feasible solution to the linear programming (LP) relaxation. Essentially, the feasibility pump can be considered as a local search procedure operating over a search neighborhood consisting of linear projections and roundings. Due to this, the feasibility pump suffers from fast convergence to local optima and the inability to escape from them. This fact was noted by the authors of the original method in Fischetti et al. (2005) and also in subsequent works (Achterberg & Berthold, 2007; Bertacco, Fischetti, & Lodi, 2007; Fischetti & Salvagnin, 2009). To circumvent this situation, several types of perturbations were proposed. Most of them occur when a cycling is detected. Although such perturbations drive the algorithm to explore other regions of the search space, they often incur information loss since the feasibility pump has no memory. For instance, some variables may take the same value across several restarts, and they could be fixed to reduce the problem size. However, the feasibility pump does not use this information.

We present a primal framework that makes use of information collected during the iterations of the feasibility pump. Such information consists of fractional infeasible solutions and their respective final roundings. The framework is able to combine the roundings that are used in subsequent feasibility pump calls. In addition, the framework also provides valuable information that can be used for variable fixing which reduces the size of LP relaxations that need to be solved. We use a biased random-key genetic algorithm to evolve a set of roundings and projections but any population-based method may be used. We restrict the presentation to pure binary and mixed binary MIPs, although the techniques presented here can be easily adapted to general MIPs.

The structure of this paper is as follows. Section 2 describes the basic idea of the feasibility pump, improvements, and variations.

* Corresponding author. Present address: AT&T Labs Research, 200 South Laurel Avenue, Middletown, NJ 07748, USA.
E-mail addresses: carlos.andrade@gatech.edu, cea@research.att.com (C.E. Andrade), shabbir.ahmed@isye.gatech.edu (S. Ahmed), george.nemhauser@isye.gatech.edu (G.L. Nemhauser), yufen.shao@exxonmobil.com (Y. Shao).

http://dx.doi.org/10.1016/j.ejor.2017.05.003
0377-2217/

In Section 3, we propose a framework used to improve the rounding and variable fixing processes. Section 4 describes the implementation details and experimental environment. Section 5 reports the computational experiments comparing the proposed framework with the feasibility pump 2.0, and a commercial solver. In Section 6, we give some conclusions.

2. The feasibility pump

2.1. Basic procedure

Here we present a high level overview of the feasibility pump. For a more detailed descriptions, see Fischetti and Salvagnin (2009) and Achterberg and Berthold (2007). Without loss of generality, consider the following mixed integer linear program

$$\text{MIP} = \min_x\{c^{\mathrm{T}}x : Ax \le b, \ell \le x \le u, x_i \in \{0,1\} \text{ for all } i \in I\},$$

where $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c, \ell, u \in \mathbb{R}^n$, and $I \subseteq N = \{1,\dots,n\}$. Let $P = \{x : Ax \le b, \ell \le x \le u\}$ be the feasible region of the LP relaxation. Consider $x^* \in P$ a fractional feasible solution for MIP (we say that x^* is LP-feasible). Let $\tilde{x}$ satisfy $\tilde{x}_i \in \mathbb{Z}$ for all $i \in I$, be an integer and not necessarily feasible solution. We compute the distance between x^* and $\tilde{x}$ using

$$\Delta(x^*, \tilde{x}) = \sum_{i\in I} |x_i^* - \tilde{x}_i|. \tag{1}$$

Using this concept of distance, we fix $\tilde{x}$ and solve the following problem:

$$\text{LP}^* = \min_{x'}\{\Delta(x', \tilde{x}) : Ax' \le b, \ell \le x' \le u, x'_i \in \mathbb{R} \text{ for all } i \in N\},$$

i.e., the original objective function is replaced by the minimization of (1) and all integrality constraints are dropped. LP* is called *Linear Program (LP) projection*. Objective function (1) can be linearized by reformulating LP*, see Fischetti and Salvagnin (2009) for details. Let x^* be an optimal solution for LP*. If x^* is integer ($x_i^* \in \mathbb{Z}$, for all $i \in I$), or $\Delta(x^*, \tilde{x}) = 0$, we have obtained an integer feasible solution and we may stop the algorithm. Otherwise, we take $\tilde{x} = \lceil x^* \rfloor$ as a *rounding* of x^* and solve the LP projection again. If during this process, we find a rounded solution already found in previous iterations, we have detected a cycle. In this case, random perturbations are performed to escape from the local minima.

A number of variations of the feasibility pump has been proposed. Bertacco et al. (2007) proposed a 3-stage algorithm where the idea is to apply pumping cycles for different sets of variables in each stage. This variation is the most used in practice. In Achterberg and Berthold (2007), a variation called *Objective Feasibility Pump* is proposed where the distance function Δ is used together with the original objective function. Other authors considered modifications during the rounding phase. In Fischetti and Salvagnin (2009), the so called Feasibility Pump 2.0 makes use of constraint propagation techniques during the rounding phase. This version is considered the most effective variation of the feasibility pump. In Baena and Castro (2011), roundings are performed over a line segment between a computed feasible point and the analytic center of the relaxed linear problem. In Boland et al. (2014), a similar approach is developed where the authors used rays directed to the feasible region instead of the analytic center. Hanafi, Lazić, and Mladenović (2010) proposed a Variable Neighborhood Pump (VNP) heuristic that combines Variable Neighborhood Branching (VNB) local search (Hansen, Mladenović, & Urošević, 2006) with the feasibility pump.

2.2. Computational burden and loss of information

The key aspect of the feasibility pump is to alternate between LP-feasible solutions obtained from a LP projection, and infeasible integer solutions obtained from roundings of non-integer solutions. Both operations can be very computationally demanding. While simple rounding is very fast (but not so effective), constraint propagation rounding can take many CPU cycles depending on the implementation details. But either rounding schemes are one or more orders of magnitude faster than the LP projection that requires solving a linear program which can be very costly. Although solvers usually keep track of the information of previous optimizations, each LP projection has a different objective function (due to the new roundings) and may change the direction of optimization making the previous optimization information useless.

In the original feasibility pump and its variants, a large amount of computational effort may be required until the algorithm finds a local minima or detects cycling. However, most information generated during this optimization is lost. Some implicit information is retained since the iterations between roundings and LP projections tend to "fix" some variables, i.e., some variables will have the same value during several pumping cycles. In this sense, all information the feasibility pump uses is very localized and immediate with respect to the last iteration.

Other factors that contribute to the information loss are the restarts. Restarts help the algorithm to explore other regions in the solution landscape. However, the way that a restart is implemented in the feasibility pump redirects the search to other regions without carrying information other than the shifting based on the fractionality. Therefore, the algorithm may "forget" the frequent values of some variables which can be used to reduce the dimensionality of the problem.

We propose to use such information by means of a evolutionary framework. The collected information is kept and evolve a pool of projections and roundings which are used to fix variables, perform local MIP searches, and create other fractional (possibly not feasible) solutions. Next, we describe this approach.

3. A primal framework for the feasibility pump

Our approach is based on a *pool* or *population* of roundings and LP projections that are evolved towards "less infeasible" solutions. In this context, a *solution* is a pair of vectors representing a rounding and a LP projection from pumping cycles, respectively. There are three main components. The first component is called the *evolutionary phase* responsible for applying the feasibility pump from several starting points and combining the results to obtain "less infeasible" solutions. The second component is called the *local MIP search phase* that uses strategies to fix and unfix variables based on the pool of solutions obtained in the evolutionary phase, and then applies an enumeration procedure. The third component is called the *fixing phase* used to reduce the dimensionality of the problem used in the evolutionary phase based on information from the pool of solutions. Next, we present the general framework and describe each phase in detail.

3.1. The general framework

Algorithm 1 describes the main procedures of our framework. The algorithm starts by preprocessing the instance and trying to reduce the domain of the variables (line 1). At this time, it also collects information from the LP relaxation that can be used to determine which type of variable fixings are to be performed and the initial percentage for the fixing phase. Dual information is collected and use to filter constraints in the local MIP search phase.

The initial pool of solutions or population is built from LP relaxations and random vectors. First in line 2, a solution to a relaxation which drops all integrality constraints is added to the population $\mathcal{P}$ (w.l.o.g., we consider a minimization problem). To include this relaxation in the initial population ensures that the algorithm will