Ten years of Feasibility Pump, and counting

Timo Berthold, Andrea Lodi, Domenico Salvagnin

August 8, 2017

Abstract

The Feasibility Pump (FP) is probably the best known primal heuristic for mixed integer programming. The original work by Fischetti, Glover, and Lodi [24], which introduced the heuristic for 0-1 mixed-integer linear programs, has been succeeded by more than twenty follow-up publications which improve the performance of the FP and extend it to other problem classes. Year 2015 was the tenth anniversary of the first FP publication. The present paper provides an overview of the diverse Feasibility Pump literature that has been presented over the last decade.

1 Introduction

The fundamental idea of all Feasibility Pump algorithms is to construct two sequences of points which hopefully converge to a feasible solution of a given optimization problem. One sequence consists of points which are feasible for a continuous relaxation (e.g., the linear programming relaxation of a mixed-integer linear programming problem, MIP in short), but possibly integer infeasible. The other sequence consists of points which are integral, but might violate some of the constraints. The next point of one sequence is always generated by minimizing the distance to the last point of the other sequence, by possibly using different distance measures in either cases (e.g., the ℓ_1- or the ℓ_2-norm).

The Feasibility Pump [1] has originally been introduced for 0-1 mixed-integer linear programs and soon been extended to MIPs. The standard way to define MIPs is in matrix and vector notation.

Definition 1.1 (mixed-integer program) *Let $m, n \in \mathbb{Z}_{\geqslant 0}$. Given a matrix $A \in \mathbb{Q}^{m \times n}$, a right-hand-side vector $b \in \mathbb{Q}^m$, an objective function vector $c \in \mathbb{Q}^n$, a lower and an upper bound vector $l \in (\mathbb{Q} \cup \{-\infty\})^n$, $u \in (\mathbb{Q} \cup \{+\infty\})^n$ and a subset $\mathcal{I} \subseteq \mathcal{N} = \{1, \ldots, n\}$, the corresponding* mixed-integer program *is given by*

$$\begin{array}{lll} \min & c^\mathsf{T} x & \\ s.t. & Ax \leqslant b & \\ & l_j \leqslant x_j \leqslant u_j & \text{for all } j \in \mathcal{N} \\ & x_j \in \mathbb{Q} & \text{for all } j \in \mathcal{N} \setminus \mathcal{I} \\ & x_j \in \mathbb{Z} & \text{for all } j \in \mathcal{I}. \end{array} \tag{1}$$

[1] Matteo Fischetti coined the name of the heuristic as a tribute to the *Electron Pump* in Isaac Asimov's novel "The Gods Themselves".

Note that the format given in Definition 1.1 is very general. First, maximization problems can be transformed to minimization problems by multiplying all objective function coefficients by -1. Similarly, "$\geqslant$" constraints can be multiplied by -1 to obtain "$\leqslant$" constraints. Equations can be replaced by two opposite inequalities. If $\mathcal{I} = \emptyset$, problem (1) is called a *linear program (LP)*. Linear programming problems assume a particular importance in the MIP technology that is based on the solution of a (possibly long) sequence of LP relaxations, i.e., the LP obtained by replacing constraints $x_j \in \mathbb{Z}, \forall j \in \mathcal{I}$ with $x_j \in \mathbb{R}, \forall j \in \mathcal{I}$ in (1). The feasible region of such a relaxation is often denoted by P.

Section 6 of this paper is dedicated to nonlinear extensions of the Feasibility Pump. Mixed-integer nonlinear programming (MINLP) is an extension of MIP that allows the objective and the constraints to be nonlinear.

Definition 1.2 (MINLP) *A* mixed-integer nonlinear program (MINLP) *is an optimization problem of the form*

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & g_i(x) \leqslant 0 & \textit{for all } i \in \mathcal{M} \\ & l_j \leqslant x_j \leqslant u_j & \textit{for all } j \in \mathcal{N} \\ & x_j \in \mathbb{Z} & \textit{for all } j \in \mathcal{I}, \end{array} \tag{2}$$

where $\mathcal{I} \subseteq \mathcal{N} \coloneqq \{1, \ldots, n\}$, $f : \mathbb{R}^n \to \mathbb{R}$, $g_i : \mathbb{R}^n \to \mathbb{R}$ *for* $i \in \mathcal{M} \coloneqq \{1, \ldots, m\}$, *and* $l \in (\mathbb{R} \cup \{-\infty\})^n$, $u \in (\mathbb{R} \cup \{+\infty\})^n$.

For practical applications, the g_i are typically assumed to be twice continuously differentiable. If the objective function f and all constraint functions $g_i, i \in \mathcal{M}$ are convex on $[l, u]$, we call (2) a *convex* MINLP to refer to the fact that its continuous relaxation is indeed convex. For disambiguity, general MINLPs are sometimes equally referred to as *nonconvex MINLPs*. Note that (2) is a MIP if and only if f and all g_i are affine functions.

The outline of the paper is as follows. In Section 2 we will outline the overall method for the mixed-integer linear case. Improvements to the different main ingredients of FP are then described in Sections 3 and 4. Theoretical aspects of the method, which are tightly connected to the perturbation mechanisms, are addressed in Section 5. Section 6 covers the many extensions of the FP scheme to the mixed-integer nonlinear case. Finally, conclusions and directions for future research are presented in Section 7.

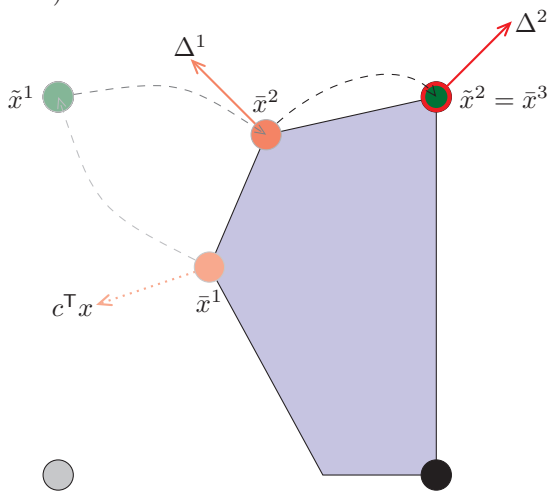
2 Feasibility Pumps for MIP

The *Feasibility Pump* algorithm was originally introduced by Fischetti, Glover, and Lodi in 2005 [24] for mixed-integer 0-1 programs, i.e., for the special case of MIPs in which $l_j = 0$ and $u_j = 1$ for all $j \in \mathcal{I}$. The main idea is as follows. The LP relaxation of a MIP is solved. The LP optimum $\bar{x}$ is then rounded to the closest integral point

$$\tilde{x} = \begin{cases} [\bar{x}_j] & \text{if } j \in \mathcal{I} \\ \bar{x}_j & \text{if } j \notin \mathcal{I}, \end{cases} \tag{3}$$

where $[\cdot]$ represents scalar rounding to the nearest integer. This part of the FP algorithm is called the *rounding step*. If $\tilde{x}$ is not feasible for the linear

Figure 1: Feasibility Pump for MIP, sequences of constraint-feasible points (red) and integer feasible points (green), original objective (dotted red) and distance functions (solid red).



constraints, the objective function of the LP is changed to the ℓ_1-norm distance function

$$\Delta(x, \tilde{x}) := \sum_{j \in \mathcal{I}} |x_j - \tilde{x}_j| = \sum_{j \in \mathcal{I}:\ \tilde{x}_j = 0} x_j + \sum_{j \in \mathcal{I}:\ \tilde{x}_j = 1} (1 - x_j) \qquad (4)$$

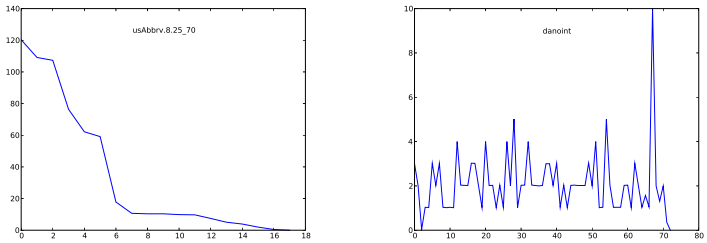
and a new LP point $\bar{x}$ is obtained by minimizing $\Delta(x, \tilde{x})$ over the linear constraints of the MIP. The process is iterated until $\tilde{x} = \bar{x}$ which implies feasibility (w.r.t. the MIP). The operation of obtaining a new $\bar{x}$ from $\tilde{x}$ is known as the *projection step*, as it consists of projecting $\tilde{x}$ to the feasible set of a continuous relaxation of the MIP along the direction $\Delta(x, \tilde{x})$. Two iterations of the algorithm are illustrated for a simple example in Figure 1, while a pseudocode description of the method is given in Figure 2.

input : MIP $\equiv \min\{c^T x : x \in P,\ x_j \text{ integer } \forall j \in \mathcal{I}\}$
output: a feasible MIP solution $\bar{x}$ (if found)

```
1 x̄ = arg min{c^T x : x ∈ P} ;
2 while not termination condition do
3 |   if x̄ is integer then return x̄;
4 |   x̃ = Round (x̄) ;
5 |   if cycle detected then Perturb (x̃);
6 |   x̄ = LinearProj (x̃) ;
7 end
```

Figure 2: Feasibility Pump—the basic scheme

The algorithm thus produces two sequences $\{\bar{x}^k\}_{k=1}^K$ and $\{\tilde{x}^k\}_{k=1}^K$ for a finite K, which is either the iteration at which a feasible solution for (1) is found or some limit set to guarantee termination. All points of the sequence $\bar{x}^k$, with k denoting the iteration count of the FP, are feasible



(a) (b)

Figure 3: Two very different FP behaviors (integrality distance vs. iterations)

for the LP relaxation, all points $\tilde{x}^k$ are integral, i.e., $\tilde{x}^k_j \in \mathbb{Z}$ for all $j \in \mathcal{I}$. Thus, $\tilde{x}^k = \bar{x}^k$ implies integrality and constraint-feasibility, which means that the corresponding point is feasible for the MIP.

The main obstacle for the original Feasibility Pump algorithm (and many of its successors) is *cycling*: after some iterations, it may happen that $\tilde{x}^k = \tilde{x}^{k'}$ with $1 \leqslant k' < k$. In this case, the procedure would enter a loop, re-visiting the sequence $\tilde{x}^{k'} \dots \tilde{x}^{k-1}$ (and $\bar{x}^{k'} \dots \bar{x}^{k-1}$) over and over again. Since the central idea of FP is to bring the sequences closely together, the risk of cycling is "naturally encoded" in the procedure and occurs very frequently in computational experiments. In the original Feasibility Pump, this issue is handled via a simple random perturbation: some of the variables in $\tilde{x}^k$ are flipped to the other bound before continuing the procedure. If the issue persists, a more aggressive perturbation akin to a *restart* is performed.

Figure 3 shows the different behavior of FP on two sample instances: while in Figure 3(a) the distance between $\bar{x}$ and $\tilde{x}$ is rapidly brought to zero without the need of any perturbation/restart, in Figure 3(b) FP exhibits a much less satisfactory behavior, with frequent restarts and perturbations that yield large oscillations of the distance function and hence reduce the probability of success of the method. It is a crucial component of many FP extensions that cycling is addressed directly, made more unlikely, or avoided completely, see more details in Section 5.

Fischetti, Glover, and Lodi demonstrated in [24] that the Feasibility Pump is very effective in finding feasible solutions, but these often are of low quality – not surprising when regarding the fact that the original objective is only considered in the very first iteration. The authors suggest to use subsequent runs of the Feasibility Pump to get better solutions. After each successful run, a primal bound constraint $c^\mathsf{T} x \leqslant \beta c^\mathsf{T} \bar{x} + (1 - \beta) c^\mathsf{T} \tilde{x}$ is added to the MIP, with $\beta \in (0, 1)$, $\bar{x}$ being an optimal solution of the original LP relaxation, and $\tilde{x}$ being the solution from the previous FP run.

Bertacco et al. [6] introduced an FP variant for mixed-integer programs with general integer variables. The authors use an auxiliary variable d_j and two auxiliary constraints for each general integer variable x_j to represent the two linear pieces of the absolute values $|x_j - \tilde{x}_j|$ whenever a variable x_j is not sitting at one of its bounds in $\tilde{x}_j$. Then, the objective function for the projection step is a modified version of function (4),

namely

$$\Delta(x,\tilde{x}) := \sum_{j\in\mathcal{I}:\ \tilde{x}_j=l_j} (x_j - l_j) + \sum_{j:\ \tilde{x}_j=u_j} (u_j - x_j) + \sum_{j\in\mathcal{I}:\ l_j<\tilde{x}_j<u_j} d_j$$

with $d_j \geqslant x_j - \tilde{x}_j$ and $d_j \geqslant \tilde{x}_j - x_j$ for all $j \in \mathcal{I}$ with $l_j < \tilde{x}_j < u_j$.

Further, the authors suggest to split the FP procedure into different stages. In stage I, the ℓ_1-norm objective function is defined only on the binary variables, if any. The auxiliary variables and constraints are only used in stage II, which also considers distances on general integers. Finally, stage III is an enumeration phase consisting of a truncated MIP search. For this, the objective function of the original MIP is replaced by the distance w.r.t. the point $\tilde{x}^k$ from the previous stages that was closest to the LP relaxation. A similar definition of the objective function was later used by Fischetti and Monaci [27] in their Proximity Search algorithm.

The main idea of the Feasibility Pump is to decompose the original problem into two problems, retaining linear and integrality constraints. Many of the extensions described below aim at solving each of the decomposed problems with an eye on the constraints of the other problem, in an attempt to accelerate convergence.

Besides enhancements of the FP for MIP itself, a natural direction of investigation is the extension of the FP idea to other contexts and/or to broader problem classes. In [1, 2], Achterberg and Gu suggest to use a FP-like algorithm, called PumpReduce, to generate alternative LP optima which can be used for improved cut generation and filtering. According to Perregaard [37], Xpress has a similar algorithm too. In [7], Berthold and Salvagnin use a similar algorithm as a basis for a branching scheme, called Cloud Branching, that is based on a set of relaxation optima. Cloud Branching applies a Feasibility Pump-like objective to the optimal face of a MIP's LP relaxation to compute multiple alternative optima, typically with reduced fractionality. These alternative optima are then used to filter potential candidate variables for which traditional branching rules such as strong branching shall be performed. A recent extension by Pal and Charkhgard [36] deals with bi-objective Pure Integer Linear Programming, where FP and local search heuristics are designed to approximately generate the so-called nondominated frontier.

Applications of FP-like algorithms to specific problems can be found in [31, 38, 13, 39, 15].

3 Improving the rounding step

Fischetti and Salvagnin [28] observed that rounding a variable can be interpreted as a temporary fixing. They suggest to propagate the minimum and maximum activities of linear constraints using these fixings as local bounds. The propagation is done via the well-known bound strengthening techniques described in [14]. By this, variables might be fixed to a value that is not the closest integer (compared to $\bar{x}^k$), in order to avoid a linear constraint becoming infeasible w.r.t. the current temporary fixings. Thus, the rounded solution might be further away from the last LP optimum than for the original FP, but it will be closer to the feasible region. In their computational experiments, the authors demonstrate that the so-called Feasibility Pump 2.0 needs fewer iterations and produces slightly

better solutions.

Baena and Castro [4] and Boland et al. [8] introduced variants of the Feasibility Pump that use integral reference points $\tilde{x}$ which are closer to the interior of the LP polyhedron. Therefore, both publications suggest to connect the LP optimum $\bar{x}$ with the analytic center [43] of the LP and search for integer points that are roundings of points on that line segment. The analytic center x^{ac} of a bounded polyhedron given in equality form ($Ax = b, x \geqslant 0$) is defined as

$$x^{\text{ac}} = \operatorname{argmin}\{-\sum_{j \in \mathcal{N}} \ln x_j \mid x > 0, Ax = b\}.$$

Baena and Castro sample points on the line segment $\bar{x} - x^{\text{ac}}$. Each of those sampled points is rounded and tested for feasibility. If none of the rounded points is feasible, a new integral reference point $\tilde{x}$ is chosen as the rounded point that minimizes the ℓ_∞-distance to the line segment point it has been rounded from. Boland et al. [8] extend this procedure by several innovative ideas. First, they observe that the set of all integral points which are roundings of some point of the line segment can be computed very efficiently. This improves the sampling step. The main overhead of the procedure in [4] lies in the computation of the analytic center in order to get a direction pointing from the LP optimum towards the interior of the polyhedron. Boland et al. suggest to use a conic combination of the normal vectors of all constraints violated by $\tilde{x}$ as a cheap heuristic approximation for a ray pointing towards the center. To the other extreme, Naoum-Sawaya [35] recently proposed a version of the Feasibility Pump with analytic centers that additionally applies a *recursive central rounding* procedure, which iteratively fixes some of the integer variables and recomputes the analytic center.

4 Improving the projection step

The main direction of modification for the projection part of the FP was the use of different objective functions for the LP. Achterberg and Berthold [3] showed a simple trick to overcome a main weakness of the FP: despite success on many instances, the produced solutions are often of poor quality. This seems natural, since the original objective of the MIP is only considered when computing the first LP solution $\bar{x}^1$. Apart from that, solution quality does not play a role, neither for the rounding nor for the projection step. The authors of [3] therefore suggest to replace function (4) by a convex combination of (4) and the original objective $c^{\mathsf{T}}x$, namely

$$\Delta_\alpha(x, \tilde{x}) := (1 - \alpha)\Delta(x, \tilde{x}) + \alpha \frac{\sqrt{|\mathcal{I}|}}{\|c\|} c^{\mathsf{T}} x$$

with $\alpha \in [0, 1]$. Here, $\|\cdot\|$ is the Euclidean norm of a vector. The convex combination factor α, and hence the influence of $c^{\mathsf{T}}x$, is reduced in every iteration. As a nice side effect, this often enables the algorithm, called Objective Feasibility Pump , to avoid cycling since the objective function $\Delta_\alpha(x, \tilde{x})$ depends on the iteration count and will be different even when the same point $\tilde{x}$ is visited more than once. As a result, the objective feasible feasibility pump finds solutions of better quality on average. However, this comes at a price of a slightly reduced success rate and increased number of iterations/running time.

Eckstein and Nediak [23] interpreted the Feasibility Pump as an implementation of a Frank-Wolfe algorithm [29], taking the ℓ_1-distance as the non-smooth concave merit function

$$\sum_{j \in \mathcal{I}} \min\{x_j, 1 - x_j\}.$$

Based on this, de Santis et al. [19, 18] interpreted and suggested the use of other concave penalty functions for non-integrality. Therefore, they weighted the different terms of the distance function with coefficients that depend on the fractionality of the corresponding variable in the last LP solution.

Boland et al. [9] use a similar penalty system, but also introduce the idea of performing several rounds of cutting plane generation to prevent the FP from cycling. This leads to fewer restarts and better and more solutions being found, but at the price of a significant increase in the complexity of the LPs (which are amended with cutting planes) being solved repeatedly, and hence in the total running time. Note that in the original Feasibility Pump, each projection step consisted of solving one LP, which is now replaced by a series of LPs. When changing the objective (distance) function in the projection phase, one can use a warm-started primal simplex algorithm to solve the LP. When amending the projection step by cutting plane generation, the additional LPs should typically be solved by a warm-started dual simplex algorithm.

Hanafi et al. [32] introduced an FP variant in which they apply a MIP search when the Feasibility Pump is about to cycle. Their variable neighborhood pump can be interpreted as applying a stage III of the Feasibility Pump variant [6] that is based on *Local Branching* [25]. The use of Local Branching to drive to feasibility partially-feasible solutions obtained by FP was already proposed in [26].

5 The perturbation step and theoretical aspects

The perturbation methods devised to avoid cycling are key ingredients of the FP method, which can greatly affect its behavior and success rate, both from the practical and theoretical point of view.

The original perturbation mechanism proposed in [24] works as follows. Whenever a cycle of length one is detected, i.e., the integer point $\tilde{x}$ obtained at the current iteration is the same as the previous one, a (small) random number, say K, of binary variables is flipped to the opposite bound, so as to minimize the increase in distance between the two sequences. This is implemented by sorting the variables by non-increasing fractionality in $\bar{x}$, and flipping the first K variables in $\tilde{x}$. The perturbation mechanism is designed to interfere as little as possible with the minimization of distance between the two sequences of points, basically exploiting the degeneracy in the distance function to pick a different integer point not too far away from $\tilde{x}$ (e.g., if all flipped variables had fractionality equal to 0.5, there would be no increase in distance at all).

If a longer cycle is detected, a different perturbation method, akin to a restart, is used, in which all binaries are possibly flipped according to some probability that depends on their fractionality.

The picture gets more complicated when general integer variables are present in the model. A careful analysis of the original source code in [6]

reveals a quite elaborated restart scheme to decide how much a single variable has to be perturbed, and how many variables have to be changed. In particular, a single variable x_j is perturbed by taking into account the size of its domain: if $u_j - l_j < M$ with a suitable large coefficient M, then the new value is picked randomly within the domain. Otherwise, the new value is picked uniformly in a large neighborhood around the lower or upper bound (if the old value is sufficiently close to one of them), or around the old value. The number of variables to be perturbed, say RP, is also very important and is changed dynamically according to the frequency of restarts, decreasing geometrically at every iteration in which restarts are not performed, and increasing linearly. Finally, RP is bounded by a small percentage (10%) of the number of general integer variables, and the variables to be changed are picked at random at every restart.

In addition, [6] also introduced a subtle random component in the rounding function. Instead of computing $\tilde{x}_j$ as $\lfloor \bar{x}_j + 0.5 \rfloor$, it is computed as $\lfloor \bar{x}_j + \tau \rfloor$, where τ is obtained as

$$\tau(\omega) = \begin{cases} 2\omega(1-\omega) & \text{if } \omega \leq \frac{1}{2} \\ 1 - 2\omega(1-\omega) & \text{if } \omega > \frac{1}{2} \end{cases}$$

where ω is a uniform random variable in $[0, 1)$.

It is worth noting that the various extensions described in the previous sections can have an influence on the need of a perturbation mechanism. In particular:

- in the objective feasibility pump there is no need to perturb the current integer point if we are in a phase in which α can still change;
- the different penalty functions described in [19, 18, 9], as well as the addition of cutting planes, can reduce the need of an explicit perturbation step;
- the rounding step based on constraint propagation described in [28] uses itself some randomization, when ranking the fractional variables.

Convergence results of the method also heavily depend on the interplay between the projection, rounding and perturbation steps. The theoretical analysis in [21] shows that the method, when applied to mixed 0-1 programs, can fail to converge if the perturbation step is limited to flip only variables that are currently fractional. At the same time, flipping variables that are currently integer in a naïve way can also lead to non-convergence and poor results. On the other hand, a perturbation method based on WalkSAT [40] is shown to yield a convergent method.

Without any perturbation, the method will still converge to a local minimizer of the distance between the two sequences (see, e.g., [9, 18, 30]), which however is not necessarily a feasible solution of the original model as the distance is usually strictly positive. The analysis in [9, 18, 30] is based on the interpretation of an *idealized* FP version, i.e., without perturbation, in terms of classical algorithms from the literature. Namely, Boland et al. [9] interpret the idealized FP as a discrete version of the *proximal point algorithm*, de Santis, Lucidi and Rinaldi [18] as a *Frank-Wolfe algorithm* applied to a suitable chosen concave and nonsmooth merit function and, finally, Geißler et al. [30] as an *alternating direction method*. In case of [30], such an interpretation allows the authors to replace the perturbation by a penalty framework, thus preserving some form of convergence that are proper of alternating direction methods.

6 Feasibility Pumps for MINLP

When designing a Feasibility Pump for MINLP, the main task is to adapt the two FP steps to nonlinearity. The two natural questions are: (i) what kind of relaxation should be solved in the projection step, and (ii) is there a different way to perform the rounding step? *Nonconvex* MINLPs represent an extra burden: even the continuous relaxation might be disconnected and is nonconvex; hence optimization over it is $\mathcal{NP}$-hard.

The first two MINLP versions of the Feasibility Pump were presented by Bonami et al. [11] and Bonami and Gonçalves [12]. Both teams of authors considered *convex* MINLPs and implemented their ideas in Bonmin [10].

The paper [12] is probably the closest to the original FP. It keeps the rounding step as in [24] and replaces solving an LP in the projection phase by solving a convex NLP, using the original distance function (4) as an objective. The perturbation scheme is less aggressive than the one of [24], flipping only a single variable.

In [11], the authors suggest using an ℓ_2-norm for the projection step. Further, their implementation of the rounding step differs significantly from all previous FP variants. Instead of performing an instant rounding to the nearest integer, they solve a MIP relaxation which is based on an outer approximation [22] of the underlying MINLP:

$$\tilde{x} = \operatorname{argmin}\{\Delta(x, \bar{x}) \mid g(\bar{x}) + J_g(\bar{x})(x - \bar{x}) \leqslant 0, x \in [l, u], x_j \in \mathbb{Z}\ \forall j \in \mathcal{I}\} \tag{5}$$

where $J_g(\bar{x})$ denotes the Jacobian of the constraint functions (summarized to a single function $g\colon \mathbb{R}^n \mapsto \mathbb{R}^m$) evaluated at the NLP optimum $\bar{x}$. Solving a MIP relaxation, despite of being an $\mathcal{NP}$-hard problem itself, is often computationally much cheaper than solving the original MINLP, see, e.g., [34]. Interestingly, the two norms have switched roles in this FP version: Where in [24] and [12], the ℓ_1-norm was used for the projection step, and the ℓ_2-norm was used for rounding, the opposite holds for [11]. An illustration of the algorithm is given in Figure 4. Note that for this FP, both steps use a distance function Δ. We denote the Manhattan distance used for the rounding step by Δ_1 and the Euclidean distance of the projection step by Δ_2, using superscripts for the iteration count.

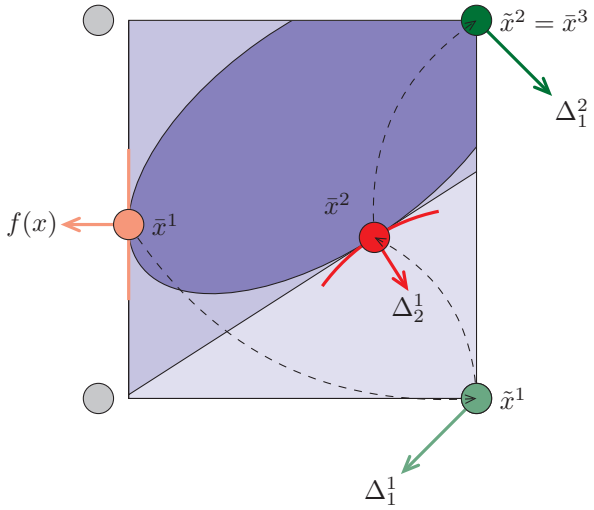
Solving (5) instead of performing a simple rounding $\tilde{x} = [\bar{x}]$ of course gives rise to better integral points (since feasibility is explicitly addressed in the rounding step), and has an important effect w.r.t. the main weakness of Feasibility Pump algorithms: cycling. For convex MINLPs, it is always possible to derive a cut

$$(\bar{x} - \tilde{x})^\mathsf{T}(x - \bar{x}) \geq 0$$

and add it to the MIP (5). By this, cycling is avoided. However, in the computational results presented in [11], the FP did not cycle even without these cuts.

To improve the solution quality of Feasibility Pumps for convex MINLP, Sharma et al. [41, 42] adapt the idea of the Objective Feasibility Pump [3] and use a combination of the original objective and an ℓ_1 distance function to extend the algorithm of Bonami and Gonçalves [12]. They also experimented with using a local linearization of the original objective at the point computed in the previous FP iteration. Li and Liu [33] modify the algorithm of Bonami et al. [11] by using approximate solutions to convex NLPs.

Figure 4: Feasibility Pump for MINLP: The truncated blue ellipsis shows the feasible region of the NLP relaxation, the light blue trapezoid depicts the MIP relaxation for the second iteration, the very light blue rectangle (which include the trapezoid) is the MIP relaxation after the first iteration. The sequence of NLP-optima is shown as red points, the sequence of MIP-optima as green points. Each red and green point also indicates for which function it is optimal: the original objective ($f(x)$, light red), a ℓ_2-distance function (Δ_2^1, dark red and bent), and ℓ_1-distance functions (Δ_1^1 and Δ_1^2, green).



Similar to the interpretation of the linear Feasibility Pump as a Frank-Wolfe algorithm [19, 18], D'Ambrosio et al. [17] gave an interpretation of a nonlinear FP as a successive projection method. The particular difficulty addressed in [16, 17] is that of handling the nonconvex NLP relaxation if adapting the algorithm of [11] to the nonconvex case. The authors suggest using a stochastic multistart approach, feeding the NLP solver with different randomly generated starting points, and solving the NLP to local optimality as if it was a convex problem. In the event that this does not lead to a feasible solution, a final NLP is solved in which the integer variables are fixed and the original objective is re-installed on the continuous variables, similar to the sub-NLP heuristic described, e.g., in [44].

Further, D'Ambrosio et al. considered solving a convex MINLP or a convex MIQP instead of an MIP in the rounding step, but gave computational evidence that this is not beneficial. To avoid cycling, their algorithm provides the MIP solver with a tabu list of previously used solutions. Linear constraints for the MIP problem are only generated from convex MINLP constraints. Finally, they showed that using an ℓ_∞-norm distance function instead of ℓ_1 as a MIP objective is competitive.

Berthold and Belotti [5] suggest three enhancements for a Feasibility Pump for nonconvex MINLP. They consider an automated selection of varying procedures for the rounding step, they take into account the Hessian of the Lagrangian as part of the distance function to take into ac-

count second order information while solving the rounding MIPs and they dynamically generate linearization cuts for nonconvex constraints during the course of the FP algorithm.

7 Conclusion

The work that followed the original FP version [24] and that has been (most likely only partially and temporarily) surveyed here gives one possible measure of the importance of FP as a research direction. According to Dey [20], it would be fair to say that a whole sub-area of work in primal heuristics is motivated by [24] and its extensions [6, 11, 17].

However, in addition to the sheer amount of follow-up publications, there is another indicator that underlines the impact of the Feasibility Pump on computational mathematical programming. There are (at least) ten MIP solvers that implement a feasibility pump as primal heuristic (CBC, CPLEX, GLPK, Gurobi, MIPCL, Mosek, SAS, SCIP, Sulum, Symphony) and (at least) four MINLP solvers (Bonmin, Couenne, Minotaur, SCIP).

What we find extremely interesting and (relatively) unexpected, is that around ten years after its first appearance, the paper has started to attract a serious bulk of theoretical work [9, 30, 21, 18]. We hope this trend, together with extensions (the latest one to bi-objective MIP [36]), practical implementations and applications to specific problems (the latest one to aircraft deconfliction [15]), will continue in the next ten years to come.

References

[1] T. Achterberg. LP basis selection and cutting planes. Presentation slides from MIP 2010 workshop in Atlanta. `https://www.isye.gatech.edu/news-events/events/past-conferences/2010-mixed-integer-programming-workshop`, 2010.

[2] T. Achterberg. LP relaxation modification and cut selection in a MIP solver. US patent, US20110131167, 2011.

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

[4] D. Baena and J. Castro. Using the analytic center in the feasibility pump. *Operations Research Letters*, 39(5):310–317, 2011.

[5] P. Belotti and T. Berthold. Three ideas for a feasibility pump for nonconvex MINLP. *Optimization Letters*, pages 1–13, 2016.

[6] L. Bertacco, M. Fischetti, and A. Lodi. A feasibility pump heuristic for general mixed-integer problems. *Discrete Optimization*, Special Issue 4(1):63–76, 2007.

[7] T. Berthold and D. Salvagnin. Cloud branching. In C. Gomes and M. Sellmann, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 7874 of *Lecture Notes in Computer Science*, pages 28–43. Springer Berlin Heidelberg, 2013.

[8] N. L. Boland, A. C. Eberhard, F. G. Engineer, M. Fischetti, M. W. P. Savelsbergh, and A. Tsoukalas. Boosting the feasibil-

ity pump, 2011. http://www.optimization-online.org/DB_FILE/2012/01/3322.pdf.

[9] N. L. Boland, A. C. Eberhard, F. G. Engineer, and A. Tsoukalas. A new approach to the feasibility pump in mixed integer programming. *SIAM Journal on Optimization*, 22(3):831–861, 2012.

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

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

[12] P. Bonami and J. Gonçalves. Heuristics for convex mixed integer nonlinear programs. *Computational Optimization and Applications*, 51:729–747, 2012.

[13] A. Bosco, D. Laganà, R. Musmanno, and F. Vocaturo. Modeling and solving the mixed capacitated general routing problem. *Optimization Letters*, 7(7):1451–1469, 2013.

[14] A. Brearley, G. Mitra, and H. Williams. Analysis of mathematical programming problems prior to applying the simplex algorithm. *Mathematical Programming*, 8:54–83, 1975.

[15] S. Cafieri and C. D'Ambrosio. Feasibility pump for aircraft deconfliction with speed regulation. Technical report, ENAC, 2017.

[16] 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.

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

[18] M. De Santis, S. Lucidi, and F. Rinaldi. A new class of functions for measuring solution integrality in the feasibility pump approach. *SIAM Journal on Optimization*, 23(3):1575–1606, 2013.

[19] M. De Santis, S. Lucidi, and F. Rinaldi. Feasibility pump-like heuristics for mixed integer problems. *Discrete Applied Mathematics*, 165:152–167, 2014.

[20] S. Dey. Personal Communication, 2017.

[21] S. Dey, A. Iroume, M. Molinaro, and D. Salvagnin. Exploiting sparsity of MILPs by improving the randomization step in feasibility pump. Technical report, Georgia Tech, 2016.

[22] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, 1986.

[23] J. Eckstein and M. Nediak. Pivot, cut, and dive: a heuristic for 0-1 mixed integer programming. *Journal of Heuristics*, 13(5):471–503, 2007.

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

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

[26] M. Fischetti and A. Lodi. Repairing MIP infeasibility through local branching. *Computers & Operations Research*, 35(5):1436–1445, 2008.

[27] M. Fischetti and M. Monaci. Proximity search for 0-1 mixed-integer convex programming. *Journal of Heuristics*, 20(6):709–731, 2014.

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

[29] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[30] B. Geißler, A. Morsi, L. Schewe, and M. Schmidt. Penalty alternating direction methods for mixed-integer optimization: A new view on feasibility pumps. Technical Report 5399, Optimization Online, 2016.

[31] O. Guyon, P. Lemaire, E. Pinson, and D. Rivreau. Near optimal and optimal solutions for an integrated employee timetabling and production scheduling problem. *IFAC Proceedings Volumes*, 42(4):1523 – 1528, 2009.

[32] S. Hanafi, J. Lazić, and N. Mladenović. Variable neighbourhood pump heuristic for 0-1 mixed integer programming feasibility. *Electronic Notes in Discrete Mathematics*, 36(0):759–766, 2010.

[33] M. Li and Q. Liu. Inexact feasibility pump for mixed integer nonlinear programming. *Information Processing Letters*, 118:110–116, 2017.

[34] R. Misener and C. A. Floudas. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming*, pages 1–28, 2012.

[35] J. Naoum-Sawaya. Recursive central rounding for mixed integer programs. *Computers & Operations Research*, 2013.

[36] A. Pal and H. Charkhgard. A feasibility pump and local search based heuristic for bi-objective pure integer linear programming. Technical Report 5902, Optimization Online, 2017.

[37] M. Perregaard. How MILP solvers can benefit from Interior Point solutions. Presentation slides from MIP 2017 workshop in Montréal. `https://sites.google.com/site/mipworkshop2017/home`, 2017.

[38] P. Pesneau, R. Sadykov, and F. Vanderbeck. Feasibility pump heuristics for column generation approaches. *11th International Symposium on Experimental Algorithms (SEA 2012)*, pages 332–343, 2012.

[39] C. Requejo and E. Santos. A feasibility pump and a local branching heuristics for the weight-constrained minimum spanning tree problem. In O. Gervasi, B. Murgante, S. Misra, G. Borruso, C. M. Torre, A. M. A. Rocha, D. Taniar, B. O. Apduhan, E. Stankova, and A. Cuzzocrea, editors, *Computational Science and Its Applications – ICCSA 2017: 17th International Conference, Trieste, Italy, July 3-6, 2017, Proceedings, Part II*, pages 669–683. Springer International Publishing, Cham, 2017.

[40] U. Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 1999.

[41] S. Sharma. Mixed-integer nonlinear programming heuristics applied to a shale gas production optimization problem. Master's thesis, Norwegian University of Science and Technology, 2013.

[42] S. Sharma, B. R. Knudsen, and B. Grimstad. Towards an objective feasibility pump for convex MINLPs. *Computational Optimization and Applications*, pages 1–17, 2014.

[43] G. Sonnevend. An "analytical centre" for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming. In A. Prékopa, J. Szelezsáan, and B. Strazicky, editors, *System Modelling and Optimization*, volume 84 of *Lecture Notes in Control and Information Sciences*, pages 866–875. Springer Berlin Heidelberg, 1986.

[44] 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. submitted.