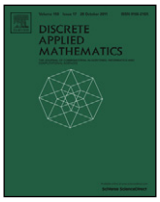
Discrete Applied Mathematics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

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



Feasibility Pump-like heuristics for mixed integer problems

M. De Santis [a], S. Lucidi [a], F. Rinaldi [b,*]

[a] Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza, Università di Roma, Via Ariosto, 25 - 00185 Roma, Italy

[b] Dipartimento di Matematica, Università di Padova, Via Trieste, 63 35121 Padua, Italy

ARTICLE INFO

Article history:
Received 22 December 2011
Received in revised form 9 June 2013
Accepted 25 June 2013
Available online xxxx

Keywords:
Mixed integer programming
Concave penalty functions
Frank–Wolfe algorithm
Feasibility problem

ABSTRACT

Finding a feasible solution to a MIP problem is a tough task that has received much attention in the last decades. The Feasibility Pump (FP) is a heuristic for finding feasible solutions to MIP problems that has encountered a lot of success as it is very efficient also when dealing with very difficult instances. In this work, we show that the FP heuristic for general MIP problems can be seen as the Frank–Wolfe method applied to a concave nonsmooth problem. Starting from this equivalence, we propose concave non-differentiable penalty functions for measuring solution integrality that can be integrated in the FP approach.

1. Introduction

Finding a first feasible solution quickly is crucial for solving Mixed Integer Programming (MIP) problems as many local-search approaches can be used only if a feasible solution is available (see e.g. [11,16]).

Several heuristic methods for finding a first feasible solution for a MIP problem have been proposed in the literature (see e.g. [4,14,18–23]). In particular, the Feasibility Pump [15] is considered one of the most efficient heuristics available.

The Feasibility Pump approach generates two sequences of points $\{\bar{x}^k\}$ and $\{\tilde{x}^k\}$ such that $\bar{x}^k$ is LP-feasible, but may not be integer feasible, and $\tilde{x}^k$ is integer, but not necessarily LP-feasible. To be more specific the algorithm starts with a solution of the LP relaxation $\bar{x}^0$ and sets $\tilde{x}^0$ equal to the rounding of $\bar{x}^0$. Then, at each iteration $\bar{x}^{k+1}$ is chosen as the nearest LP-feasible point in the ℓ_1-norm to $\tilde{x}^k$, and $\tilde{x}^{k+1}$ is obtained as the rounding of $\bar{x}^{k+1}$. The idea of the algorithm is to reduce at each iteration the distance between the points of the two sequences, until the two points are the same and an integer feasible solution is found. Unfortunately, it can happen that the distance between $\bar{x}^{k+1}$ and $\tilde{x}^k$ is greater than zero and $\tilde{x}^{k+1} = \tilde{x}^k$, and the strategy can stall. In order to overcome this drawback, random perturbations and restart procedures are performed.

Various papers devoted to further improvements of the Feasibility Pump have been developed. In [5], the authors extended the Feasibility Pump in order to handle MIP problems with both 0–1 and integer variables and they further exploited the FP information to drive a subsequent enumeration phase. Achterberg and Berthold [1] proposed a different distance function which takes into account the original objective function in order to improve the quality of the feasible solution found. Some efforts have also been made to propose new rounding techniques [3,6,17]. In [3,6] the main idea is that of choosing a rounded solution along a line segment passing through the LP-feasible solution. In [17], an exact method combining a diving-like procedure and constraint propagation is proposed.

* Corresponding author. Tel.: +39 049 827 1424.
E-mail addresses: mdesantis@dis.uniroma1.it (M. De Santis), stefano.lucidi@dis.uniroma1.it (S. Lucidi), rinaldi@math.unipd.it (F. Rinaldi).

0166-218X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.dam.2013.06.018

For the case of 0–1 MIPs an interesting interpretation of the FP has been given by Eckstein and Nediak in [14]: they noticed that the FP heuristic may be seen as a form of Frank–Wolfe procedure applied to a nonsmooth merit function. This concave nonsmooth function penalizes the violation of the integrality constraints but its reduction might not correspond to a decrease in the number of variables that violate integrality. Starting from this interpretation in [12] the relationship between the Feasibility Pump and the Frank–Wolfe algorithm has been exploited to define a new version of the Feasibility Pump obtained by applying the Frank–Wolfe algorithm to a different nonsmooth concave function which penalizes the violation of the integrality constraints and has the good property that its reduction cannot correspond to an increase in the number of fractional variables.

In this work, we first show that the equivalence between the FP and the Frank–Wolfe algorithm still holds for the general integer case. Then we extend the results proposed in [12] for 0–1 MIPs to general mixed integer problems. The extension is not straightforward. In the 0–1 case, due to the fact that the integer feasible points lie on the boundary of the relaxed feasible set, we can use a suitable class of concave penalty functions and a Frank–Wolfe based approach to find an integer feasible solution. The first choice depends on the fact that, when minimizing a concave function over a polyhedron, the global optima, if any, are on the boundary of the polyhedron. On the other hand, the choice of the Frank–Wolfe approach is motivated by the fact that it is well suited for the class of problems we want to solve, since in this case the algorithm moves from a vertex to another until it finds a stationary point. When dealing with general MIP problems, there are integer feasible points that are in the interior of the feasible set. So, the direct extension of the approach considered in [12] would lead to penalty functions that are not concave anymore (the functions admit global minima in the interior of the feasible set). This would further imply a loss in the efficiency of the Frank–Wolfe algorithm, as we should include a suitable line search technique to guarantee the convergence of the method. In order to overcome these issues, we need to exploit the hidden concavity of the penalty problem obtained with the approach described in [12] by considering a new equivalent concave problem having a larger number of variables.

The paper is organized as follows. In Section 2, we give a brief review of the Feasibility Pump heuristic for general MIP problems. In Section 3, we show the equivalence between the FP heuristic and the Frank–Wolfe algorithm applied to a nonsmooth merit function. In Section 4, we introduce new nonsmooth merit functions for dealing with general integer variables, and discuss their properties. We present our algorithm in Section 5, and in Section 6 we explain how it can be integrated in the Objective Feasibility Pump [1]. In Section 7 we give a performance comparison of our algorithm with the Objective FP showing that the merit functions proposed can improve the efficiency of the FP approach in terms of CPU time.

In the following, given a concave function $f : R^n \to R$, we denote by $\partial f(x)$ the set of supergradients of f at the point x, namely

$$\partial f(x) = \{v \in R^n : f(y) - f(x) \le v^T(y - x), \forall y \in R^n\}.$$

2. The Feasibility Pump heuristic for general MIP problems

We consider a MIP problem of the form:

$$\begin{aligned} &\min\ c^T x \\ &\text{s.t. } Ax \ge b \\ &l \le x \le u \\ &x_j \in \mathbb{Z}, \quad \forall j \in I, \end{aligned} \tag{MIP}$$

where $A \in \mathbf{R}^{m\times n}$ and $I \subseteq \{1, 2, \dots, n\}$ is the set of indices of the integer variables. Let $P = \{x : Ax \ge b, l \le x \le u\}$ denote the polyhedron of the LP-relaxation of (MIP). The Feasibility Pump [5,15] starts from the solution of the LP relaxation problem $\bar{x}^0 := \arg\min\{c^T x : x \in P\}$ and generates two sequences of points $\bar{x}^k$ and $\tilde{x}^k$: $\bar{x}^k$ is LP-feasible, but may be integer infeasible; $\tilde{x}^k$ is integer, but not necessarily LP-feasible. At each iteration $\bar{x}^{k+1} \in P$ is the nearest point in ℓ_1-norm to $\tilde{x}^k$:

$$\bar{x}^{k+1} := \arg\min_{x\in P} \Delta(x, \tilde{x}^k) \tag{1}$$

where

$$\Delta(x, \tilde{x}^k) = \sum_{j\in I} |x_j - \tilde{x}^k_j|.$$

The point $\tilde{x}^{k+1}$ is obtained as the rounding of $\bar{x}^{k+1}$. The procedure stops if at some iteration h, $\bar{x}^h$ is integer or, in case of failing, if it reaches a time or iteration limit. In order to avoid stalling issues and loops, the Feasibility Pump performs a perturbation step. Here we report a brief outline of the basic scheme [5,15]:

```
The Feasibility Pump (FP) for general MIPs - basic version

Initialization: Set $k = 0$, let $\bar{x}^0 := \arg\min\{c^T x : x \in P\}$
While (not stopping condition) do

    Step 1 If ($\bar{x}^k$ is integer) return $\bar{x}^k$
    Step 2 Compute $\tilde{x}^k = round(\bar{x}^k)$
    Step 3 If (cycle detected) $perturb(\tilde{x}^k)$
    Step 4 Compute $\bar{x}^{k+1} := \arg\min\{\Delta(x, \tilde{x}^k) : x \in P\}$
    Step 5 Update $k = k + 1$

End While
```

Now we give a better description of the rounding and the perturbing procedures used respectively at *Step* 2 and at *Step* 3 [5,15]:

Round: This function transforms a given point $\bar{x}^k$ into an integer one, $\tilde{x}^k$. The easiest choice is that of rounding each component $\bar{x}^k_j$ with $j \in I$ to the nearest integer, while leaving the continuous components of the solution unchanged. Formally,

$$\tilde{x}^k_j = \begin{cases} [\bar{x}^k_j] = \lfloor \bar{x}^k_j + \tau \rfloor & \text{if } j \in I \\ \bar{x}^k_j & \text{otherwise} \end{cases} \tag{2}$$

where $\tau = 0.5$, and $\lfloor \cdot \rfloor$ represents the floor function (a function which maps a real number to the largest previous integer). Another possibility is that of using a random τ like that described in [5]:

$$\tau(\omega) = \begin{cases} 2\omega(1-\omega), & \text{if } \omega \le \dfrac{1}{2} \\ 1 - 2\omega(1-\omega), & \text{otherwise} \end{cases} \tag{3}$$

where ω is a uniform random variable in $[0, 1)$. Using the definition (3), threshold τ can assume a value between 0 and 1, but values close to 0.5 are more likely than those close to 0 or 1.

As already mentioned in the introduction, alternative rounding techniques have been recently proposed (see [3,6,17] for further details).

Perturb: The aim of the perturbation procedure is to avoid cycling and it consists of two heuristics. To be more specific:

– if $\tilde{x}^k_j = \tilde{x}^{k+1}_j$ for all $j \in I$ a weak perturbation is performed, namely, a random number of integer constrained components, chosen as to minimize the increase in the distance $\Delta(\bar{x}^{k+1}, \tilde{x}^{k+1})$, is moved using the following rule:

$$\tilde{x}^{k+1}_j = \begin{cases} \tilde{x}^k_j + 1, & \text{if } \bar{x}^{k+1}_j > \tilde{x}^k_j \\ \tilde{x}^k_j - 1, & \text{if } \bar{x}^{k+1}_j < \tilde{x}^k_j. \end{cases} \tag{4}$$

– A *restart* operation, consisting of random perturbation of some entries of $\tilde{x}^{k+1}$, is performed if one of the following situations occur:
  – the point $\tilde{x}^{k+1}$ is equal, in its integer components, to a previously generated point;
  – the distance $\Delta(\bar{x}^{k+1}, \tilde{x}^{k+1})$ did not decrease by at least 10% in the last KK iterations.

The Feasibility Pump for general MIPs described in [5] consists of three different stages. In the first stage (binary stage), a few iterations (so-called pumping rounds) are performed just on the binary variables $B \subseteq I$ by relaxing the integrality conditions on the general integer variables. If this does not yield to a feasible solution, a second stage starts from a point $\tilde{x}$ visited in Stage 1 which was closest to the LP polyhedron. In the second stage, the integrality condition on all (binary and non-binary) variables is restored, and the FP method continues by taking into account all the integrality constraints (this requires the introduction, at each iteration, of the additional variables needed to express the distance function with respect to the current point). If still no solution is found, a third stage is executed. Using a point $\tilde{x}$ from Stage 2 closest to P, the MIP

$$\min\{\Delta(x, \tilde{x}) | x \in P, x_j \in \mathbb{Z}, \forall j \in I\}$$

is processed by a MIP solver which stops after the first feasible solution is found.

3. The FP heuristic and the Frank–Wolfe method

While for the 0–1 case the equivalence between the FP heuristic and the Frank–Wolfe method has been pointed out first in [14] and then further investigated and exploited in [12]; the relationship between FP and the Frank–Wolfe method in the general integer case is by far less clear [9,10]. The aim of this section is that of clarifying this equivalence showing that the

FP heuristic without any perturbation (i.e. without Step 3) can be interpreted as the Frank–Wolfe algorithm with unitary stepsize applied to a concave, non-differentiable merit function.

At Step 4 of the FP, problem (1) has to be solved. This problem can be written as (see [5]):

$$\begin{aligned} \min \; & \sum_{j\in I:\tilde{x}_j^k=l_j}(x_j-l_j)+\sum_{j\in I:\tilde{x}_j^k=u_j}(u_j-x_j)+\sum_{j\in I:l_j<\tilde{x}_j^k<u_j}d_j \\ \text{s.t.}\; & Ax\ge b \\ & l\le x\le u \\ & -d_j\le x_j-\tilde{x}_j^k\le d_j\quad \forall j\in I: l_j<\tilde{x}_j^k<u_j, \end{aligned} \tag{5}$$

where the variables d_j are introduced to model the nonlinear function $d_j=|x_j-\tilde{x}_j^k|$ for integer variables x_j that are not equal to one of their bounds in the rounded solution $\tilde{x}^k$.

In order to see the equivalence with the Frank–Wolfe method we briefly recall the unitary stepsize Frank–Wolfe method for concave non-differentiable functions. Let us consider the problem

$$\begin{aligned} &\min f(x) \\ & x\in P \end{aligned} \tag{6}$$

where $P\subset R^n$ is a non-empty polyhedral set that does not contain lines going to infinity in both directions, $f: R^n\to R$ is a concave, non-differentiable function, bounded below on P.

The Frank–Wolfe algorithm with unitary stepsize (see [26,25] for further details) at each iteration k produces a new point

$$x^{k+1}=\arg\min_{x\in P}\,(g^k)^T x$$

where $g^k\in\partial f(x^k)$. Then, the algorithm involves only the solution of linear programming problems, and it is proved in [26] that it converges to a stationary point $x^\star$ in a finite number of iterations.

In the 0–1 case the Feasibility Pump can be seen as the Frank–Wolfe algorithm applied for the minimization of the following objective function

$$\phi(x)=\sum_{j\in I}\min\{x_j,1-x_j\}$$

over the relaxed feasible set [12]. In the general integer case the same reasoning would lead to the solution of the following minimization problem

$$\begin{aligned} \min \; & \sum_{j\in I}\min\{x_j-l_j,u_j-x_j,|x_j-s_{1j}|,\ldots,|x_j-s_{m_j j}|\} \\ \text{s.t.}\; & Ax\ge b \\ & l\le x\le u \end{aligned} \tag{7}$$

where for each $j\in I$ we have $\{s_{ij}: i=1,\ldots,m_j\}=(l_j,u_j)\cap Z$. Unlike the 0–1 case, the objective function of problem (7) is not concave, thus the unitary stepsize Frank–Wolfe method would no more be able to converge to a stationary point in a finite number of iterations and a suitable line search technique to guarantee the convergence should be used within the method. However, by enlarging the variable space [5], problem (7) can be transformed in the following equivalent concave problem

$$\begin{aligned} \min \; & \sum_{j\in I}\min\{x_j-l_j,u_j-x_j,\hat{d}_{1j},\ldots,\hat{d}_{m_j j}\} \\ \text{s.t.}\; & Ax\ge b \\ & l\le x\le u \\ & -\hat{d}_{ij}\le x_j-s_{ij}\le \hat{d}_{ij}\quad \forall j\in I, i=1,\ldots,m_j. \end{aligned} \tag{8}$$

Given the point $\tilde{x}_j^k$, one iteration of the Frank–Wolfe algorithm applied to problem (8), consists in the solution of the following LP-problem

$$\begin{aligned} \min \; & \sum_{j\in I:\tilde{x}_j^k\le l_j+\frac{1}{2}}x_j-\sum_{j\in I:\tilde{x}_j^k>u_j-\frac{1}{2}}x_j+\sum_{j\in I:l_j+\frac{1}{2}<\tilde{x}_j^k\le u_j-\frac{1}{2}}\hat{d}_{i_j j} \\ \text{s.t.}\; & Ax\ge b \\ & l\le x\le u \\ & -\hat{d}_{i_j j}\le x_j-s_{i_j j}\le \hat{d}_{i_j j}\quad \forall j\in I, \end{aligned} \tag{9}$$

where for each $j \in I$ the index $i_j \in \{1, \dots, m_j\}$ is such that

$$\hat{d}^k_{i_j j} = \min_{i=1,\dots,m_j} \hat{d}^k_{ij}. \tag{10}$$

We notice that Problem (9) is a reformulation of Problem (5). Once Problem (9) has been solved, we set

$$\hat{d}^{k+1}_{ij} = |\bar{x}^{k+1}_j - s_{ij}|, \quad \forall j \in I.$$

This step is crucial for the equivalence, as it is needed to ensure that at each iteration of the Frank–Wolfe method we have

$$s_{i_j j} = [\bar{x}^k_j]. \tag{11}$$

Now, taking into account (11) and naming variable $\hat{d}_{i_j j}$ as d_j we get Problem (5) which is the LP Problem solved at Step 4 of the Feasibility Pump.

Example 1. Consider the following problem of one dimension:

$$\begin{aligned} &\min\ cx \\ &\text{s.t. } x \in \{0, 1, 2\}. \end{aligned} \tag{12}$$

The use of the FP heuristic (without any perturbation) for finding a feasible solution of Problem (12) is equivalent to apply the Frank–Wolfe method with unitary stepsize to the following problem

$$\begin{aligned} &\min\{x, 2 - x, \hat{d}\} \\ &\text{s.t. } x \in [0, 2] \\ &-\hat{d} \le x - 1 \le \hat{d}. \end{aligned} \tag{13}$$

In Fig. 1 we visualize, on the left, the graph of the objective function of Problem (13) and on the right, the contour lines of the objective function and the additional constraint $-\hat{d} \le x - 1 \le \hat{d}$.

4. Nonsmooth merit functions for solving general MIPs

In the previous section, we have seen that the Feasibility Pump for general MIP problems is equivalent to the Frank–Wolfe algorithm applied to the following problem

$$\begin{aligned} &\min f(x, \hat{d}) = \sum_{j \in I} \min\{x_j - l_j, u_j - x_j, \hat{d}_{1j}, \dots, \hat{d}_{m_j j}\} \\ &\text{s.t. } Ax \ge b \\ &l \le x \le u \\ &-\hat{d}_{ij} \le x_j - s_{ij} \le \hat{d}_{ij} \quad \forall j \in I, i = 1, \dots, m_j \end{aligned} \tag{14}$$

where $\hat{d} = (\hat{d}_{11}, \dots, \hat{d}_{m_1 1}, \hat{d}_{12}, \dots, \hat{d}_{m_2 2}, \dots, \hat{d}_{1N}, \dots, \hat{d}_{m_N N})$ and $N = |I|$.

In general, there is no relationship between the reduction of the objective function $f(x, \hat{d})$ and the reduction in the number of variables that violate integrality. This can be easily seen from the following example.

Example 2. Let us consider the following two points $x, y \in [0, 2]^4$:

$$x = \left(0, \frac{3}{2}, 0, 2\right)^T; \qquad y = \left(0, \frac{7}{6}, \frac{7}{6}, 2\right)^T.$$

It is easy to notice that

$$\sum_{j=1}^{4} \min\{y_j, 2 - y_j, |y_j - 1|\} = \frac{1}{3} < \frac{1}{2} = \sum_{j=1}^{4} \min\{x_j, 2 - x_j, |x_j - 1|\},$$

but the number of fractional components of y is greater than the number of fractional components of x.

As we have already noticed in [12], it would be better to use a function having the following features:

(i) it decreases whenever the number of integer variables increases;
(ii) if it decreases, then the number of fractional variables does not increase.

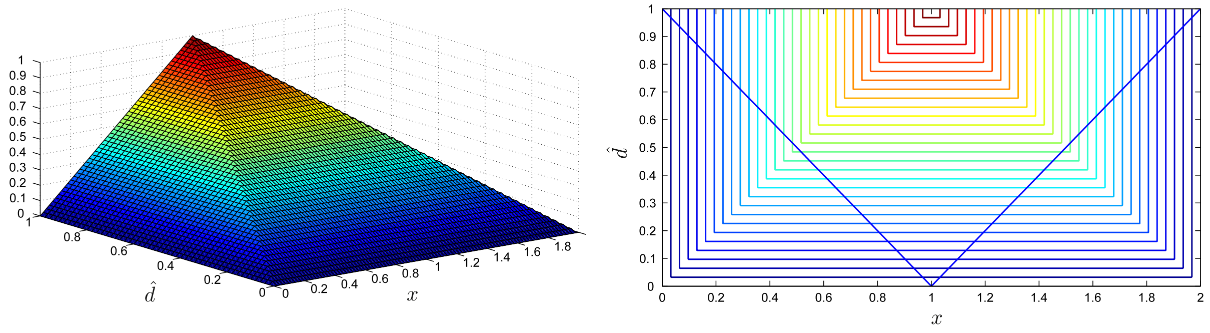


Fig. 1. Problem (13).

In this context, a good choice would be the following:

$$\psi(x) = card\{x_j : j \in I, x_j \notin Z \cap [l_j, u_j]\}. \tag{15}$$

The function (15) can be rewritten as:

$$\psi(x) = \sum_{j \in I} \sigma(\min\{x_j - l_j, u_j - x_j, |x_j - s_{j1}|, \dots, |x_j - s_{jm_j}|\}) \tag{16}$$

where $\sigma : \mathbf{R} \to \mathbf{R}^+$ is the *step function*:

$$\sigma(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

and $\{s_{ij} : i = 1, \dots, m_j\} = (l_j, u_j) \cap Z$.

Obviously minimizing function (16) over a polyhedral set is a very tough problem, as the step function is nonconvex and discontinuous. In this context, it would be useful to define approximations of function (16) that are easier to handle from a computational point of view and guarantee satisfaction of (i) and (ii) when evaluated on the vertices of a polyhedron. In order to do that, we generalize the results in [12] to the general MIP case. We first introduce the following two scalar functions:

Definition 1. For all $i \in I$, let $\rho_i : \mathbb{R} \to \mathbb{R}_+$ be a function such that

$$\rho_i(t) = 0 \quad \text{if } t \in (l_i, u_i) \cap Z;$$
$$\rho_i(t) > 0 \quad \text{if } t \notin (l_i, u_i) \cap Z.$$

Definition 2. Let $\varphi_\epsilon : \mathbb{R}_+ \to \mathbb{R}$ be a function that verifies the following properties: for all $\gamma > 0$, a value $\bar{\epsilon} > 0$ and a value $M > 0$ exist such that

(i) for $\tilde{\alpha} > \gamma$ we have

$$\varphi_\epsilon(0) - \varphi_\epsilon(\tilde{\alpha}) \leq -M; \tag{17}$$

(ii) for $\bar{\alpha}, \tilde{\alpha} > \gamma$ we have

$$|\varphi_\epsilon(\bar{\alpha}) - \varphi_\epsilon(\tilde{\alpha})| \leq \frac{M}{n} \tag{18}$$

for all $\epsilon \in (0, \bar{\epsilon}]$.

The first function penalizes the fact that a scalar is fractional and the second one can be seen as an approximation of the step function. By combining the two functions, we get the following separable function:

$$\phi_\epsilon(x) = \sum_{i \in I} \varphi_\epsilon(\rho_i(x_i)), \tag{19}$$

that can be considered a suitable approximation of function (16).

Proposition 1. *Let $V \subset [l, u]^n$ be the set of vertices of a polytope $P = \{x : Ax \geq b, x \in [l, u]\}$.*

Let $\phi_\epsilon(x)$ defined as in (19).

Then, for $x, y \in V$:

(a) *$\psi(x) < \psi(y)$ implies $\phi_\epsilon(x) < \phi_\epsilon(y)$;*
(b) *$\phi_\epsilon(x) < \phi_\epsilon(y)$ implies $\psi(x) \le \psi(y)$.*

Proof. Let $\widetilde{V} \subset V$ be the set of the fractional vertices of P.
Let $x \in \widetilde{V}$, we define the set $\widetilde{I}(x)$ as

$$\widetilde{I}(x) = \{i \in I : x_i \notin Z\}.$$

We define

$$\rho^\star = \min_{x \in \widetilde{V}} \min_{i \in \widetilde{I}(x)} \rho(x_i),$$

where $\rho(\cdot)$ is the function described in Definition 1.
Let $\bar{\epsilon} > 0$ be the threshold value such that the properties (17), (18) of $\varphi_\epsilon(\cdot)$ in Definition 2 hold, when $\gamma = \rho^\star$.
In the sequel of the proof we assume $\epsilon \in (0, \bar{\epsilon}]$.

(a) We first start considering two vertices x, y of the polytope P satisfying the following condition:

$$\psi(x) < \psi(y). \tag{20}$$

We notice that y is not integer feasible, otherwise $\psi(x) = \psi(y) = 0$. Therefore, we can define the set of indices related to its fractional components:

$$W = \{j \in \{1, \dots, n\} \mid j \in I, \rho_j(y_j) \neq 0\}.$$

Since y is not integer feasible, $W \neq \emptyset$. Then we can write the function ϕ_ϵ as follows:

$$\phi_\epsilon(y) = \sum_{j \in I} \varphi_\epsilon(\rho_j(y_j)) = \sum_{j \in W} \varphi_\epsilon(\rho_j(y_j)) + \sum_{j \in I \setminus W} \varphi_\epsilon(\rho_j(y_j)).$$

Similarly we define the set of indices related to the fractional components of x:

$$U = \{i \in \{1, \dots, n\} \mid i \in I, \rho_i(x_i) \neq 0\},$$

and we write

$$\phi_\epsilon(x) = \sum_{i \in I} \varphi_\epsilon(\rho_i(x_i)) = \sum_{i \in U} \varphi_\epsilon(\rho_i(x_i)) + \sum_{i \in I \setminus U} \varphi_\epsilon(\rho_i(x_i)).$$

It is easy to see that, by (20), we have

$$|U| < |W| \quad \text{and} \quad |I \setminus U| > |I \setminus W|.$$

Now, we consider two different cases:
(i) $|W| - |U| = 1$:
We can assume that there exists an index $\bar{j}$ such that

$$W \setminus \{\bar{j}\} = U \quad \text{and} \quad (I \setminus U) \setminus \{\bar{j}\} = I \setminus W.$$

Then we can write:

$$\phi_\epsilon(x) = \varphi_\epsilon(\rho_{\bar{j}}(x_{\bar{j}})) + \sum_{\substack{j \in W \\ j \neq \bar{j}}} \varphi_\epsilon(\rho_j(x_j)) + \sum_{\substack{j \in I \setminus U \\ j \neq \bar{j}}} \varphi_\epsilon(\rho_j(x_j)) \tag{21}$$

and

$$\phi_\epsilon(y) = \varphi_\epsilon(\rho_{\bar{j}}(y_{\bar{j}})) + \sum_{\substack{j \in W \\ j \neq \bar{j}}} \varphi_\epsilon(\rho_j(y_j)) + \sum_{\substack{j \in I \setminus U \\ j \neq \bar{j}}} \varphi_\epsilon(\rho_j(y_j)). \tag{22}$$

By (21) and (22) and the fact that $\rho_j(x_j) = \rho_j(y_j) = 0$ for all $j \in (I \setminus U) \setminus \{\bar{j}\}$, we obtain:

$$\begin{aligned}\phi_\epsilon(x) - \phi_\epsilon(x) &= \varphi_\epsilon(\rho_{\bar{j}}(x_{\bar{j}})) - \varphi_\epsilon(\rho_{\bar{j}}(y_{\bar{j}})) + \sum_{\substack{j \in W \\ j \neq \bar{j}}} (\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))) \\ &\le \varphi_\epsilon(\rho_{\bar{j}}(x_{\bar{j}})) - \varphi_\epsilon(\rho_{\bar{j}}(y_{\bar{j}})) + \sum_{\substack{j \in W \\ j \neq \bar{j}}} |\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))|. \end{aligned} \tag{23}$$

From Definition 1, we have that $\rho_{\bar{j}}(x_{\bar{j}}) = 0$, $\rho_{\bar{j}}(y_{\bar{j}}) > \rho^\star$, $\rho_j(x_j) > \rho^\star$ and $\rho_j(y_j) > \rho^\star$ for all $j \in W \setminus \{\bar{j}\}$. Then, by (17) and (18):

$$\begin{aligned}\phi_\epsilon(x) - \phi_\epsilon(y) &\le \varphi_\epsilon(\rho_{\bar{j}}(x_{\bar{j}})) - \varphi_\epsilon(\rho_{\bar{j}}(y_{\bar{j}})) + \sum_{\substack{j\in W\\ j\neq\bar{j}}} |\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))| \\ &\le -M + (|I|-1)\frac{M}{n} < 0. \qquad (24)\end{aligned}$$

Thus obtaining

$$\phi_\epsilon(x) < \phi_\epsilon(y).$$

(ii) $|W| - |U| > 1$.
In this case, we can assume that there exists a set $\bar{J}$ such that

$$W \setminus \{\bar{J}\} = U \quad \text{and} \quad (I \setminus U) \setminus \{\bar{J}\} = I \setminus W.$$

Then we can write

$$\phi_\epsilon(x) = \sum_{j\in\bar{J}} \varphi_\epsilon(\rho_j(x_j)) + \sum_{j\in U} \varphi_\epsilon(\rho_j(x_j)) + \sum_{\substack{j\in I\setminus U\\ j\notin\bar{J}}} \varphi_\epsilon(\rho_j(x_j))$$

and

$$\phi_\epsilon(y) = \varphi_\epsilon(\rho_j(y_j)) + \sum_{\substack{j\in W\\ j\notin\bar{J}}} \varphi_\epsilon(\rho_j(y_j)) + \sum_{j\in I\setminus W} \varphi_\epsilon(\rho_j(y_j)).$$

By the same reasoning as before, we obtain:

$$\phi_\epsilon(x) - \phi_\epsilon(y) \le \sum_{j\in\bar{J}} (\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))) + \sum_{j\in W\setminus\{\bar{J}\}} |\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))|.$$

Now we notice that $\rho_j(x_j) = 0$, $\rho_j(y_j) > \rho^\star$ for all $j \in \bar{J}$ and $\rho_j(x_j) > \rho^\star$, $\rho_j(y_j) > \rho^\star$ for all $j \in W \setminus \{\bar{J}\}$. Then, by using (17) and (18), we have

$$\begin{aligned}\phi_\epsilon(x) - \phi_\epsilon(y) &\le \sum_{j\in\bar{J}} (\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))) + \sum_{j\in W\setminus\{\bar{J}\}} |\varphi_\epsilon(\rho_j(x_j)) - \varphi_\epsilon(\rho_j(y_j))| \\ &\le -M|\bar{J}| + (|I| - |\bar{J}|)\frac{M}{n} < 0,\end{aligned}$$

thus obtaining

$$\phi_\epsilon(x) < \phi_\epsilon(y).$$

(b) We prove point (b) by contradiction. Let us assume that there exist two vertices x, y of the polytope P such that $\phi_\epsilon(x) < \phi_\epsilon(y)$ and

$$\psi(x) > \psi(y). \qquad (25)$$

By (25), recalling the first part of the proof, we have that $\phi_\epsilon(x) > \phi_\epsilon(y)$. This contradicts our initial assumption. □

For what concern the function $\rho(\cdot)$ a straightforward choice can be the following: for all $i \in I$,

$$\rho_i(x_i) = \min\{x_i - l_i, u_i - x_i, |x_i - s_{i1}|, \dots, |x_i - s_{im_i}|\}.$$

As regarding function $\varphi_\epsilon(\cdot)$, taking into account Definition 2 and the ideas developed in [24,27,28], we consider the following terms:

Logarithmic function

$$\varphi_\epsilon(t) = \ln(t+\epsilon) \qquad (26)$$

Hyperbolic function

$$\varphi_\epsilon(t) = -(t+\epsilon)^{-p} \qquad (27)$$

Exponential function

$$\varphi_\epsilon(t) = 1 - \exp\left(-\frac{1}{\epsilon}t\right) \qquad (28)$$

Logistic function

$$\varphi_\epsilon(t) = \left[1 + \exp\left(-\frac{1}{\epsilon} t\right)\right]^{-1} \tag{29}$$

where $\epsilon, p > 0$.

In the following we prove that for a particular choice of the φ_ϵ term, the properties (17), (18) in Definition 2 are satisfied.

Proposition 2. *For the term (26), there exists a value $\bar{\varepsilon} > 0$ such that for any $\varepsilon \in (0, \bar{\varepsilon}]$ properties (17), (18) in Definition 2 are satisfied.*

Proof. Let $\gamma > 0$. As the function $\varphi_\epsilon(t)$ is strictly increasing, for any choice of $\tilde{\alpha}, \bar{\alpha} > \gamma$ we have

$$|\varphi_\epsilon(\tilde{\alpha}) - \varphi_\epsilon(\bar{\alpha})| \le \varphi_\epsilon(1) - \varphi_\epsilon(\gamma).$$

We set

$$M = n(\varphi_\epsilon(1) - \varphi_\epsilon(\gamma)),$$

therefore

$$|\varphi_\epsilon(\tilde{\alpha}) - \varphi_\epsilon(\bar{\alpha})| \le \frac{M}{n}. \tag{30}$$

Let us define $\bar{\varepsilon} > 0$ such that for all $\varepsilon \in (0, \bar{\varepsilon})$

$$\ln(\varepsilon) - \ln(\gamma + \varepsilon) \le -M. \tag{31}$$

From (30) and (31) properties (17), (18) of Definition 2 follow. □

The result proved in Proposition 2 for the term (26) can also be proved for the terms (27)–(29) repeating the same arguments, thus all the merit functions proposed are suitable to penalize the variables that violate the integrality constraints.

Summarizing, given an approximation $\phi_\epsilon(x)$ defined as in (19) with the $\rho_i(\cdot)$ and the $\varphi_\epsilon(\cdot)$ proposed, we can solve, in place of the original FP problem (8), the following problem

$$\begin{aligned} &\min\ \phi_\epsilon(x) = \sum_{j\in I} \varphi_\epsilon(\rho_j(x_j)) \\ &\text{s.t. } Ax \ge b \\ &l \le x \le u; \end{aligned} \tag{32}$$

where $\{s_{ij} : i = 1, \dots, m_j\} = (l_j, u_j) \cap Z$. As it has been done in Section 3, we enlarge the variable space and transform problem (32) to the following equivalent concave problem

$$\begin{aligned} &\min \sum_{j\in I} \varphi_\epsilon(\min\{x_j - l_j, u_j - x_j, \hat{d}_{1j}, \dots, \hat{d}_{m_j j}\}) \\ &\text{s.t. } Ax \ge b \\ &l \le x \le u \\ &-\hat{d}_{ij} \le x_j - s_{ij} \le \hat{d}_{ij} \quad \forall j \in I, i = 1, \dots, m_j. \end{aligned} \tag{33}$$

Taking into account the monotonicity of the functions φ_ϵ (26)–(29), we notice that problem (33) can be rewritten as

$$\begin{aligned} &\min f_p(x, d) = \sum_{j\in I} \min\{\varphi_\epsilon(x_j - l_j), \varphi_\epsilon(u_j - x_j), \varphi_\epsilon(\hat{d}_{1j}), \dots, \varphi_\epsilon(\hat{d}_{m_j j})\} \\ &\text{s.t. } Ax \ge b \\ &l \le x \le u \\ &-\hat{d}_{ij} \le x_j - s_{ij} \le \hat{d}_{ij} \quad \forall j \in I, i = 1, \dots, m_j. \end{aligned} \tag{34}$$

The Frank–Wolfe algorithm is used for solving the minimization problem (34). This choice is mainly due to the fact that the algorithm, at each step, moves from a vertex to another guaranteeing the reduction of the chosen approximation. Therefore, Proposition 1 ensures that, at each iteration, the number of the fractional components of the current solution does not increase.

5. The reweighted Feasibility Pump for MIPs

As in [12], the use of the φ_ϵ functions (26)–(29) leads to a new FP scheme in which the ℓ_1-norm used for calculating the next LP-feasible point is replaced with a "weighted" ℓ_1-norm of the form

$$\Delta_W(x,\tilde{x}) = \sum_{j\in I:\tilde{x}_j^k=l_j} w_j(x_j-l_j) + \sum_{j\in I:\tilde{x}_j^k=u_j} w_j(u_j-x_j) + \sum_{j\in I:l_j<\tilde{x}_j^k<u_j} w_j d_j \tag{35}$$

where the variables $d_j = |x_j - \tilde{x}_j|$ satisfy the constraints

$$-d_j \le x_j - \tilde{x}_j \le d_j \quad \forall j\in I : l_j < \tilde{x}_j < u_j, \tag{36}$$

and $w_j, j = 1,\ldots,n$ are positive weights depending on the φ_ϵ term chosen. Here we report an outline of the algorithm:

Reweighted Feasibility Pump (RFP) for general MIPs - basic version

```
Initialization: Set $k = 0$, let $\bar{x}^0 := \arg\min\{c^Tx : x\in P\}$
While (not stopping condition) do
    Step 1 If ($\bar{x}^k$ is integer) return $\bar{x}^k$
    Step 2 Compute $\tilde{x}^k = round(\bar{x}^k)$
    Step 3 If (cycle detected) $perturb(\tilde{x}^k)$
    Step 4 Compute $\bar{x}^{k+1} := \arg\min\{\|W^k(x-\tilde{x}^k)\|_1 : x\in P\}$
    Step 5 Update $k = k + 1$
End While
```

We assume that the *round* procedure is the same as that described in Section 2 for the original version of the FP heuristic [15,5]. As regards the *perturb* procedure, we first perturb the point $\tilde{x}^k$ using the same procedure as that described in Section 2, then for all indices $j\in I$ such that $\tilde{x}^{k+1}\neq\tilde{x}^k$, we add 0.5 to $\bar{x}_j^{k+1}$ (if $\tilde{x}_j^{k+1} > \tilde{x}_j^k$) or subtract 0.5 to $\bar{x}_j^{k+1}$ (if $\tilde{x}_j^{k+1} < \tilde{x}_j^k$). Anyway, different rounding and perturbing procedures can be suitably developed. In particular the scheme based on constraint propagation proposed in [17] can be integrated in the RFP.

Following the reasoning of Section 3, we can reinterpret the reweighted FP heuristic without perturbation as the unitary stepsize Frank–Wolfe algorithm applied to the merit function f_p. Let us now consider a generic iteration k of the reweighted FP. At Step 4, the algorithm computes the solution of the LP problem

$$\begin{aligned}
\min \ & \sum_{j\in I:\bar{x}_j^k\le l_j+\frac{1}{2}} w_j^k x_j - \sum_{j\in I:\bar{x}_j^k> u_j-\frac{1}{2}} w_j^k x_j + \sum_{j\in I:l_j+\frac{1}{2}<\bar{x}_j^k\le u_j-\frac{1}{2}} w_j^k d_j \\
\text{s.t. } & Ax\ge b \\
& l\le x\le u \\
& -d_j \le x_j - [\bar{x}_j^k] \le d_j \quad \forall j\in I : l_j+\frac{1}{2} < \bar{x}_j^k \le u_j - \frac{1}{2}.
\end{aligned} \tag{37}$$

As in Section 3, we can prove that problem (37) is equivalent to

$$\begin{aligned}
\min \ & \sum_{j\in I:\bar{x}_j^k\le l_j+\frac{1}{2}} \tilde{w}_j^k x_j - \sum_{j\in I:\bar{x}_j^k> u_j-\frac{1}{2}} \tilde{w}_j^k x_j + \sum_{j\in I,\ i=1,\ldots,m_j:l_j+\frac{1}{2}<\bar{x}_j^k\le u_j-\frac{1}{2}} \tilde{w}_{ij}^k \hat{d}_{i_j j} \\
\text{s.t. } & Ax\ge b \\
& l\le x\le u \\
& -\hat{d}_{i_j j} \le x_j - s_{ij} \le \hat{d}_{i_j j} \quad \forall j\in I,
\end{aligned} \tag{38}$$

with $s_{i_j j} = [\bar{x}_j^k]$.

By setting

$$\tilde{w}_j^k = |g_{x_j}^k|$$
$$\tilde{w}_{ij}^k = |g_{\hat{d}_{i_j j}}^k|$$

with $g^k \in \partial f_p(\bar{x}^k, \bar{d}^k)$, and $\bar{d}^k_{i_j j} = |x_j - s_{i_j j}|, \forall j \in I$, problem (38) can be seen as the iteration of the Frank–Wolfe method with unitary stepsize applied to the following minimization problem

$$\begin{aligned} &\min f_p(x, d) = \sum_{j\in I} \min\{\varphi_\epsilon(x_j - l_j), \varphi_\epsilon(u_j - x_j), \varphi_\epsilon(\hat{d}_{1j}), \ldots, \varphi_\epsilon(\hat{d}_{m_j j})\} \\ &\text{s.t. } Ax \geq b \\ &l \leq x \leq u \\ &-\hat{d}_{ij} \leq x_j - s_{ij} \leq \hat{d}_{ij} \quad \forall j \in I, i = 1, \ldots, m_j. \end{aligned} \tag{39}$$

In order to better understand the meaning of the Reweighted Feasibility Pump we give an example.

Example 3. By choosing φ_ϵ equal to the logarithmic function, we can write problem (39) as follows:

$$\begin{aligned} &\min f_p(x, \hat{d}) = \sum_{j\in I} \min\{\log(x_j - l_j + \epsilon), \log(u_j - x_j + \epsilon), \log(\hat{d}_{1j} + \epsilon), \ldots, \log(\hat{d}_{m_j j} + \epsilon)\} \\ &\text{s.t. } Ax \geq b \\ &l \leq x \leq u \\ &-\hat{d}_{ij} \leq x_j - s_{ij} \leq \hat{d}_{ij} \quad \forall j \in I, i = 1, \ldots, m_j. \end{aligned} \tag{40}$$

Then, at an iteration k of the Reweighted Feasibility Pump heuristic, we have

$$w_j^k = \begin{cases} \tilde{w}_j^k = \dfrac{1}{\bar{x}_j^k - l_j + \epsilon} & \text{if } \bar{x}_j^k \leq l_j + 1/2 \\ \tilde{w}_j^k = \dfrac{1}{u_j - \bar{x}_j^k + \epsilon} & \text{if } \bar{x}_j^k \geq u_j - 1/2 \\ \tilde{w}_{ij}^k = \dfrac{1}{\bar{d}^k_{i_j j} + \epsilon} & \text{if } s_{i_j j} - 1/2 \leq \bar{x}_j^k \leq s_{i_j j} + 1/2. \end{cases} \tag{41}$$

Example 4. Consider the problem (12) of Example 1. The use of the RFP heuristic (without any perturbation) for finding a feasible solution of Problem (12) is equivalent to apply the Frank–Wolfe method with unitary stepsize to the following problem

$$\begin{aligned} &\min\{\varphi_\epsilon(x), \varphi_\epsilon(2 - x), \varphi_\epsilon(\hat{d})\} \\ &\text{s.t. } x \in [0, 2] \\ &-\hat{d} \leq x - 1 \leq \hat{d} \end{aligned} \tag{42}$$

where function φ_ϵ is chosen among terms (26)–(29). In Fig. 2 we visualize, on the left, the graph of the objective function of Problem (42) with φ_ϵ chosen as the Exponential function and on the right, the contour lines of the objective function and the additional constraint $-\hat{d} \leq x - 1 \leq \hat{d}$.

6. Adding the original objective function to the FP scheme

One of the targets we have when we use a heuristic like the Feasibility Pump on a MIP problem is that of finding a high-quality solution, which means to find a feasible point for the problem with the objective function $c^T x$ as small as possible. A drawback of the original Feasibility Pump scheme is that the quality of the solutions in terms of the objective value often tends to be poor. This is easily explained by the fact that the original scheme discards the original objective function of the problem after the first iteration. Different approaches have been developed for dealing with this issue. Bertacco, Fischetti and Lodi in [5] make use of improvement heuristics based on local search, such as local branching [16] or RINS [11]. In [1] a different approach, called Objective Feasibility Pump, is developed by Achterberg and Berthold. The idea is that of gradually reducing the influence of the objective function and increasing the weight of the artificial objective function of the Feasibility Pump, so that the algorithm concentrates its search on the region of high-quality points. In particular, the objective function of the LPs is a convex combination of the original objective function with the distance function $\Delta(x, \tilde{x})$:

$$\Delta_\theta(x, \tilde{x}) = \frac{1-\theta}{\|\Delta\|}\Delta(x, \tilde{x}) + \frac{\theta}{\|c\|}c^T x$$

where $\|\Delta\| = \sqrt{|I|}$ and $\theta \in [0, 1]$. At each iteration k, coefficient θ^k is geometrically decreased by a factor $\nu < 1$ (i.e. $\theta^{k+1} = \nu\theta^k$). The introduction of the new function further requires a modification of the cycle detection step. While in the original scheme if we visit the same integer point twice we can be sure that we are in a cycle, this is not the case in the

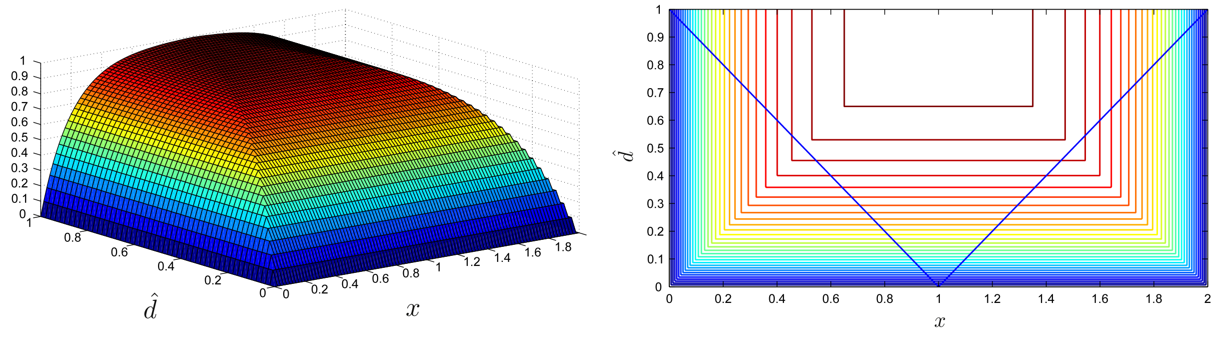


Fig. 2. Problem (42).

Table 1
Comparison between Obj FP and Obj RFP (geometric means).

	Obj FP		Obj Exp, $1/\epsilon = 0.5$		Obj Logis, $1/\epsilon = 0.1$	
	Iter	Time	Iter	Time	Iter	Time
All instances	20.6516	2.0746	12.4590	1.7896	22.1848	2.0558
MIPLIB instances	35.4144	1.977	25.1857	1.9491	38.9995	1.9587
COR@L instances	18.7410	2.0926	10.9765	1.7623	20.0426	2.0737

modified scheme, because the objective function Δ_θ has changed in the meantime. We therefore store, at each iteration k, the pair $(\tilde{x}^k, \theta^k)$ and a cycle is detected if there exist two iterations k_i and k_j, with $k_i < k_j$, such that $\tilde{x}^{k_i} = \tilde{x}^{k_j}$ and $\theta^{k_i} - \theta^{k_j} \leq \delta_\theta$, where $\delta_\theta \in [0, 1]$ is a fixed parameter (see [1] for further details). This approach can be easily adapted to the Reweighted Feasibility Pump, where the new objective function of the LPs becomes:

$$\Delta_{W,\theta}(x, \tilde{x}) = \frac{1-\theta}{\|\Delta\|} \Delta_W(x, \tilde{x}) + \frac{\theta}{\|c\|} c^T x. \tag{43}$$

7. Numerical results

In this section, we describe the details of the computational experiments we have performed. The aim is to analyze the possible effects of the merit functions proposed on the Feasibility Pump scheme. In order to do that, we suitably modified the Objective Feasibility Pump (Obj FP) described in [1] to include the new functions, and compared this new version of the code, namely the Objective Reweighted Feasibility Pump (Obj RFP), with the original one.

The test set used consists of 403 instances from MIPLIB2003 [2] and COR@L libraries [7]. The algorithms were implemented in C++ and we used ILOG Cplex 12.0.0 [8] as solver of the linear programming problems. All tests have been run on an Intel Core2 3.00 GHz with 16 GB of RAM.

The numerical experience reported in [12] for MIP 0–1 instances showed that, among the penalty terms proposed, the Exponential function and the Logistic function give the best results, and the RFP algorithm obtained using this two penalty terms is competitive with the original FP. Hence, we decided to analyze the behavior of both the Exponential function (Exp) and the Logistic function (*Logis*) when embedded in the objective RFP (Obj RFP) defined in Section 6, and how they compare with the Obj FP [1]. In particular, in the Obj RFP-Exp the distance $\Delta_W(x, \tilde{x})$ is defined using the term (28) with $1/\epsilon = 0.5$ and in the Obj RFP-Logis the distance $\Delta_W(x, \tilde{x})$ is defined using the term (29) with $1/\epsilon = 0.1$. The choice of the merit function parameters is critical for the efficiency of the algorithm and the values $1/\epsilon = 0.5$ for the Exp term and $1/\epsilon = 0.1$ for the Logis term represent a good compromise between theory and practice. Indeed, from Proposition 1, it would be better to set the parameter of a chosen merit function to a sufficiently small value. However, if the parameter is very small, numerical difficulties arise, as the slope of the graph of the function to be minimized gets very large when approaching integer values. For the Obj FP we set the parameters to the default values used in [5] enabling the use of the modified objective function as proposed in [1]. In order to better assess the effectiveness of the new approach, we ran the algorithms with stage 3 disabled and we compared the number of iterations and the CPU time needed to find a first feasible solution and the quality of the feasible solution found. The Obj FP and the Obj RFP-Logis failed to find a feasible solution on 90 problems while Obj RFP-Exp failed on 81 problems.

We first analyze the average results of the various algorithms. In Table 1, the geometric means of the iterations and the CPU time needed for each algorithm to find a first feasible solution are reported. In the calculations of the geometric means individual values smaller than 1 are replaced by 1. The reported results seem to confirm that the Exp and Logis RFP algorithms are competitive with the FP algorithm.

To give a further interpretation of the results of the various algorithms in terms of number of iterations and CPU time needed to find a first feasible solution, we decided to use performance profiles [13]. Given a set A of n_a solvers and a set P of

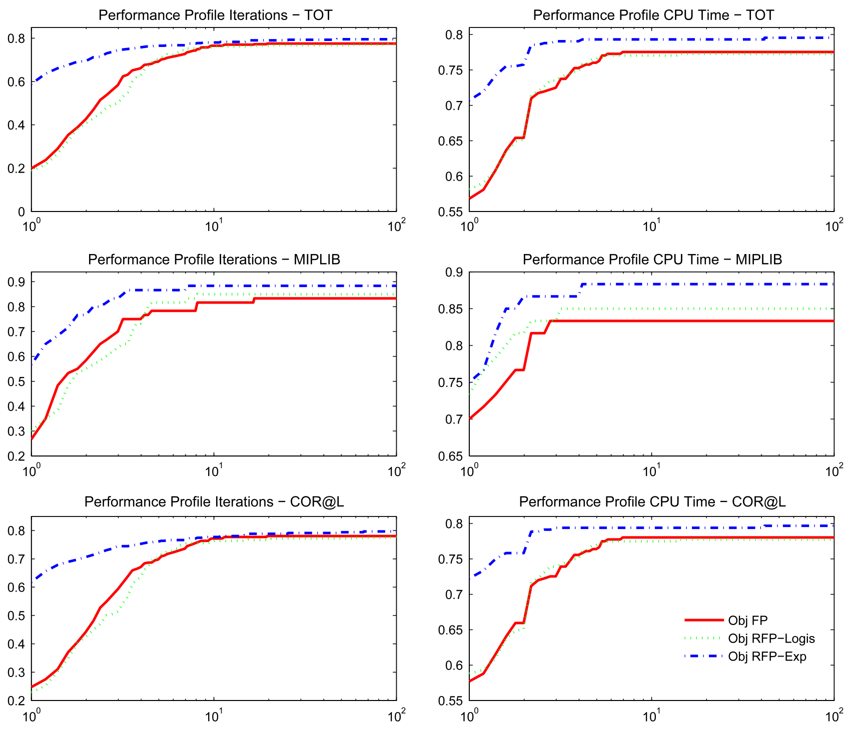


Fig. 3. Comparison between Obj FP, Obj RFP-Logis and Obj RFP-Exp.

n_p problems we compare the performance on a performance measure $m_{p,a}$ (e.g. number of iteration, CPU time) on problem $p \in P$ by algorithm $a \in A$ with the best performance by any algorithm on this problem using the following *performance ratio*

$$r_{p,a} = \frac{m_{p,a}}{\min\{m_{p,a} : a \in A\}}.$$

Then, we obtain an overall assessment of the performance of the algorithm by defining the following value

$$\rho_a(\tau) = \frac{1}{n_p}\mathrm{size}\{p \in P : r_{p,a} \le \tau\},$$

which represents the probability for algorithm $a \in A$ that the performance ratio $r_{p,a}$ is within a factor $\tau \in R$ of the best possible ratio. The function ρ_a represents the distribution function for the performance ratio. Thus $\rho_a(1)$ gives the fraction of problems for which the algorithm a was the most effective, $\rho_a(2)$ gives the fraction of problems for which the algorithm a is within a factor of 2 of the best algorithm, and so on.

In Fig. 3, we report the performance profiles related to the comparison between Obj FP, Obj RFP-Exp and Obj RFP-Logis in terms of number of iterations and CPU time. We indicate with TOT, MIPLIB and CORAL the performance profiles related respectively to all problems, MIPLIB problems and CORAL problems. In the *x-axis*, τ is reported in logarithmic scale. By taking a look at Fig. 3 we can notice that there are large gaps in the beginning of the performance diagrams between Obj RFP-Exp and the other two. This means that Obj RFP-Exp has a higher number of wins both in terms of number of iterations and in terms of computational time. Furthermore the Obj RFP-Exp profile always dominates the other two. The Obj FP and the Obj RFP-Logis have very similar profiles both for the number of iterations and for the computational time.

Now, we want to analyze the behavior of the algorithms in terms of solution quality. To this aim, for all MIP instances such that an integer feasible solution is found by all the algorithms, we consider the gap from the best known solution. Given a problem $p \in P$ and an algorithm $a \in A$ we denote by $sol(p)$ the best known solution of problem p and by $obj(p, a)$ the objective function evaluated in the integer feasible solution found by algorithm a for problem p. We define the gap of the solution found by algorithm a for the problem p as

$$gap(p, a) = \frac{|sol(p) - obj(p, a)|}{|sol(p)|}.$$

In the computation of the gap, we excluded any instance p such that $sol(p) = 0$, or for which at least an algorithm fails to find an integer feasible solution. We mention that, in our test set, 292 instances are considered for the gap computation. Over these 292 instances the Obj FP found the best (optimal) solution for 30 (19) instances, the Obj RFP-Logis for 31 (22) instances and the Obj RFP-Exp for 25 (15) instances. We further report the number of wins (minimum CPU time and minimum Gap)

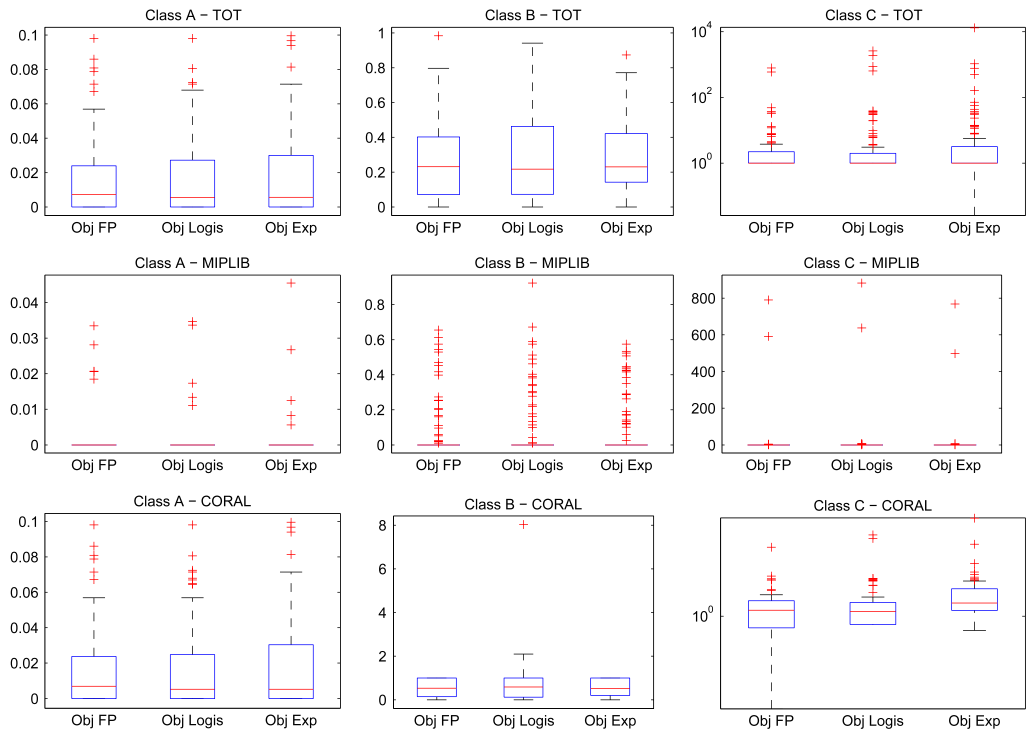


Fig. 4. Box plots of the GAP.

for the three algorithms: the Obj FP has 116 number of wins, the Obj RFP-Logis 100 and the Obj RFP-Exp 123. We report, in Table 2, the detailed results for a meaningful subset of instances over the 292 considered for the gap computation.

In order to better assess the differences in terms of the gap among the Obj FP, the Obj RFP-Logis and the Obj RFP-Exp, we considered the 292 problems for which the gap was computable and we divided the problems in three different classes:

- *Class* A. Problems for which all the algorithms found a solution with a gap between 0% and 10% (76 problems);
- *Class* C. Problems for which any algorithm found a solution with a gap greater than 100% (98 problems);
- *Class* B. All the problems that are neither in Class A nor in Class C (118 problems).

We report in Fig. 4 the box plots related to the distribution of the gap for the Obj FP, the Obj RFP-Logis and the Obj RFP-Exp on the three classes of problems. We indicate with TOT, MIPLIB and COR@L the plots related respectively to all problems, MIPLIB problems and COR@L problems. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers (i.e. single values that do not belong to the intervals drawn) are plotted individually.

By taking a look at the results, we can see that, for what concerns the quality of the solution, the three algorithms are comparable. Furthermore, we would like to notice that the introduction of the objective function, similarly to [1], significantly improves the quality of the solutions found by the RFP approach.

Finally, we want to highlight that the proposed functions can be included within other FP variants like e.g. [17] where the rounding scheme is replaced by a scheme based on constraint propagation, or [3,6] where a new integer feasible point is obtained by a procedure that examines rounded solutions along a given line segment. Anyway, the performance analysis of these approaches is beyond the scope of this paper.

8. Conclusions

In this paper, we considered the problem of finding a first feasible solution to a mixed integer linear programming problem. We exploited the relationship between the Feasibility Pump and the Frank–Wolfe algorithm to define a modified version of the Feasibility Pump obtained applying the Frank–Wolfe algorithm to new nonsmooth concave functions. These functions penalize the violation of the integrality constraints and have the good property that their reduction cannot correspond to an increase in the number of fractional variables. Due to this property, the functions proposed should speed up the convergence towards integer feasible points. We reported computational results on a set of 403 mixed integer linear programming problems. This numerical experiments indicate that the new version of the Feasibility Pump obtained by using the exponential term compares favorably with the Objective FP in terms of CPU time and it is comparable with respect to the quality of the solutions found. Finally, we think that the merit functions proposed, together with the original FP merit

Table 2
Detailed results for some meaningful instances.

	Obj FP			Obj RFP-Logis			Obj RFP-Exp		
	Gap %	Iter	Time	Gap %	Iter	Time	Gap %	Iter	Time
aflow30a	2.5	40	1	0.9	29	0	1.4	31	0
dano3-3	0.0	12	16	0.0	25	33	0.0	6	15
dano3-4	0.0	33	33	0.0	33	43	0.0	9	17
dano3-5	0.0	21	45	0.0	36	57	0.0	12	37
mcf2	0.1	107	2	0.0	80	1	0.1	160	2
neos12	0.0	22	8	0.0	12	5	0.0	3	3
neos-574665	2.3	267	0	0.1	41	0	1.0	59	0
neos-584866	1.0	59	55	1.0	40	102	1.0	36	18
neos-612162	0.0	11	1	0.0	24	1	0.0	7	0
neos-702280	1.0	7	77	1.0	6	51	1.0	4	63
neos-738098	0.0	68	6	0.0	69	9	0.0	47	8
neos-777800	0.0	97	21	0.0	57	9	0.0	48	4
neos-839859	6.4	43	0	3.6	43	0	3.8	16	0
neos-872648	1.0	33	20	1.0	33	24	1.0	31	29
neos-873061	1.0	29	21	1.0	29	17	1.0	29	20
neos-911880	2.8	79	0	2.3	66	0	1.1	33	0
neos-935627	1.0	6	27	1.0	9	18	1.0	2	4
neos-936660	1.0	6	21	1.0	7	15	1.0	2	4
neos-937446	1.0	6	17	1.0	13	8	1.0	2	4
neos-937511	1.0	6	11	1.0	13	10	1.0	2	5
neos-937815	1.0	6	25	1.0	6	12	1.0	2	5
neos-941262	1.0	17	19	1.0	6	10	1.0	2	4
neos-941313	1.0	16	24	1.0	6	17	1.0	23	48
neos-948126	1.0	17	26	1.0	11	10	1.0	2	5
neos-948268	0.0	22	14	0.0	26	14	0.0	15	9
neos-983171	1.0	17	14	1.0	9	11	1.0	2	4
neos-984165	1.0	19	15	1.0	11	11	1.0	2	4
neos-1121679	48.4	67	0	38.9	85	0	30.2	60	0
neos-1420205	7.6	217	1	0.0	29	0	0.3	29	0
neos-1420546	1.0	42	11	1.0	42	10	1.0	9	3
neos-1430811	1.0	110	109	1.0	99	86	1.0	60	67
neos-1603965	1.0	105	23	1.0	74	19	1.0	23	4
neos-1622252	0.0	26	3	0.1	17	5	0.0	7	1
markshare1	790.0	67	0	638.0	85	0	498.0	60	0
markshare2	592.0	70	0	882.0	85	0	768.0	59	0
ramos3	1.0	8	253	1.0	8	341	1.0	5	187

function, can be fruitfully used in a parallel framework and/or in a multistart strategy to define algorithms that use more than one function at time.

References

[1] T. Achterberg, T. Berthold, Improving the feasibility pump, Discrete Optimization 4 (2007) 77–86.
[2] T. Achterberg, T. Koch, A. Martin, MIPLIB 2003, Operations Research Letters 34 (2006) 361–372. Problems available at http://miplib.zib.de.
[3] D. Baena, J. Castro, Using the analytic center in the feasibility pump, Operations Research Letters 39 (2011) 310–317.
[4] E. Balas, S. Schmieta, C. Wallace, Pivot and shifta mixed integer programming heuristic, Discrete Optimization 1 (1) (2004) 3–12.
[5] L. Bertacco, M. Fischetti, A. Lodi, A feasibility pump heuristic for general mixed-integer problems, Discrete Optimization 4 (2007) 63–76.
[6] N.L. Boland, A.C. Eberhard, F.G. Engineer, M. Fischetti, M.W.P. Savelsbergh, A. Tsoukalas, Boosting the feasibility pump, Report C-OPT 2012-03, The University of Newcastle, Callaghan, NSW, 2308, Australia, 2012.
[7] COR@L, COR@L MIP library, http://coral.ie.lehigh.edu/data-sets/mixed-integer-instances/.
[8] ILOG, Cplex, http://www.ilog.com/products/cplex.
[9] C. D'ambrosio, Application-oriented mixed integer nonlinear programming, Ph.D. Thesis, University of Bologna, 2009.
[10] C. D'ambrosio, Application-oriented mixed integer nonlinear programming, 4OR 8 (3) (2010) 319–322.
[11] E. Danna, E. Rothberg, C. Le Pape, Exploring relation induced neighborhoods to improve MIP solution, Mathematical Programming 102 (1) (2005) 71–90.
[12] M. De Santis, S. Lucidi, F. Rinaldi, A new class of functions for measuring solution integrality in the feasibility pump approach, SIAM Journal on Optimization (2013) http://dx.doi.org/10.1137/110855351 (in press).
[13] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profile, Mathematical Programming 91 (2002) 201–213.
[14] J. Eckstein, M. Nediak, Pivot, cut, and dive: a heuristic for 0–1 mixed integer programming, Journal of Heuristics 13 (2007) 471–503.
[15] M. Fischetti, F. Glover, A. Lodi, The feasibility pump, Mathematical Programming 104 (2005) 91–104.
[16] M. Fischetti, A. Lodi, Local branching, Mathematical Programming 98 (1–3) (2003) 23–47.
[17] M. Fischetti, D. Salvagnin, Feasibility pump 2.0, Mathematical Programming Computation 1 (2009) 201–222.
[18] F. Glover, M. Laguna, General purpose heuristics for integer programming part I, Journal of Heuristics 3 (1997).
[19] F. Glover, M. Laguna, General purpose heuristics for integer programming part II, Journal of Heuristics 3 (1997).
[20] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publisher, Boston, Dordrecht, London, 1997.
[21] F. Glover, A. Løkketangen, D.L. Woodruff, Scatter search to generate diverse MIP solutions, in: M. Laguna, J. Gonzàlez-Velarde (Eds.), OR Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research, Kluwer Academic Publishers, 2000, pp. 299–317.
[22] F.S. Hillier, Efficient heuristic procedures for integer linear programming with an interior, Operations Research 17 (1969) 600–637.

[23] F.S. Hillier, R.M. Saltzman, A heuristic ceiling point algorithm for general integer linear programming, Management Science 38 (2) (1992) 263–283.
[24] S. Lucidi, F. Rinaldi, Exact penalty functions for nonlinear integer programming problems, Journal of Optimization Theory and Applications 145 (2010) 479–488.
[25] O.L. Mangasarian, Machine learning via polyhedral concave minimization, in: H. Fischer, B. Riedmueller, S. Schaeffler (Eds.), Applied Mathematics and Parallel Computing-Festschrift for Klaus Ritter, Physica, Heidelberg, 1996, pp. 175–188.
[26] O.L. Mangasarian, Solutions of general linear complementarity problems via nondifferentiable concave minimization, Acta Mathematica Vietnamica 22 (1) (1997) 199–205.
[27] F. Rinaldi, New results on the equivalence between zero–one programming and continuous concave programming, Optimization Letters 3 (3) (2009) 377–386.
[28] F. Rinaldi, F. Schoen, M. Sciandrone, Concave programming for minimizing the zero-norm over polyhedral sets, Computational Optimization and Applications 46 (2010) 467–486.