Applied Mathematics and Computation 217 (2010) 25–41

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

A critical review of discrete filled function methods in solving nonlinear discrete optimization problems

Siew Fang Woon [a,b,*], Volker Rehbock [a]

[a] Department of Mathematics and Statistics, Curtin University of Technology, Bentley, Western Australia 6102, Australia
[b] Physical Science, College of Arts and Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia

ARTICLE INFO

Keywords:
Discrete filled function
Discrete global optimization
Nonlinear discrete optimization
Heuristic

ABSTRACT

Many real life problems can be modeled as nonlinear discrete optimization problems. Such problems often have multiple local minima and thus require global optimization methods. Due to high complexity of these problems, heuristic based global optimization techniques are usually required when solving large scale discrete optimization or mixed discrete optimization problems. One of the more recent global optimization tools is known as the discrete filled function method. Nine variations of the discrete filled function method in literature are identified and a review on theoretical properties of each method is given. Some of the most promising filled functions are tested on various benchmark problems. Numerical results are given for comparison.

1. Introduction

Many real life applications, such as production planning, finance, scheduling, and operations involve integer valued decision variables. We distinguish between discrete optimization problems, where all decision variables are integer valued, and mixed discrete optimization problems, where only some of the decision variables have integer values. The latter type are often decomposed into purely discrete and continuous subproblems, respectively, and hybrid algorithms for their solutions are developed on this basis. The discrete parts of these hybrid algorithms are similar in nature to the purely discrete algorithms, which we address in this paper. Most practical discrete and mixed discrete optimization problems are nonlinear and known to have more than one locally optimal solutions. This suggests the need for global optimization techniques which seek the best solution amongst multiple local optima. Global optimization problems may be unconstrained or constrained, and different algorithms have been developed, depending on whether constraints are present as well as on the nature of these constraints.

The challenge in global optimization is to avoid being trapped in the basins surrounding local minimizers. Several global methods have been proposed for solving discrete optimization problems. These techniques can be classified into two main categories: exact methods and heuristic methods. The branch and bound method [18,20,21], the cutting plane method [7,10,45], Lagrangian relaxation [9,13], the nonlinear Lagrangian relaxation method [40,41], the discrete Lagrangian method [38,39], dynamic programming [22], and relaxation techniques [3,17,29] are popular exact methods. These exact methods can ensure that a global solution is found when solving small size discrete optimization problems. However, such methods require excessive computational time when solving large scale problems. Furthermore, only well-structured problems with good analytical properties can be solved efficiently using these exact methods.

* Corresponding author at: Department of Mathematics and Statistics, Curtin University of Technology, Bentley, Western Australia 6102, Australia.
E-mail addresses: woonsiewfang@yahoo.com (S.F. Woon), rehbock@maths.curtin.edu.au (V. Rehbock).

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.05.009

Since nonlinear discrete optimization problems are generally NP-hard, there are no efficient exact algorithms with polynomial-time complexity for solving them. Hence, a heuristic computational approach is required, especially for high-dimensional problems. The heuristic methods include greedy-search [1,4,6,8], simulated annealing [24,30], genetic algorithm [2,34,37], tabu search [14,23], and filled function techniques. Though these methods cannot guarantee a global solution, satisfactory results can often be found for high-dimensional nonlinear discrete optimization problems in a reasonable amount of computational time.

The discrete filled function method is one of the more recently developed global optimization tools for discrete optimization problems. Once a local minimum has been determined by an ordinary descent method, the discrete filled function approach introduces an auxiliary function to avoid entrapment in the basin associated with this minimum. The local minimizer of the original function becomes a local maximizer of the auxiliary function. By minimizing the auxiliary function, the search moves away from the current local minimizer in the hope of escaping the basin associated with this minimizer. Note that the auxiliary function is defined in terms of one or more parameters and needs to possess certain properties, details of which are discussed in Section 3. The first filled function was introduced by Ge in the late 1980s [11] in the context of solving continuous global optimization problems. In [12], Ge and Huang extended the continuous filled function concept to solve nonlinear discrete optimization problems, where a continuous global optimization problem is formulated to approximate the discrete global optimization problem, before solving it by the continuous filled function method. When a global minimizer of the continuous approximation is found, the nearest integer point is used to approximate the global solution of the discrete problem. However, the approximating continuous optimization problem always generates more local minimizers than the original discrete one, thus making it more difficult to determine a global solution. Numerical results reported in [26] have shown that the true global minimizer is difficult to determine using this approach. A detailed analysis of the continuous filled function can also be found in [26].

Zhu [46] is believed to be the first researcher to introduce a discrete equivalent of the continuous filled function method in late 1990s. Such an approach is now known as a discrete filled function method or discrete global descent method. A discrete filled function method is able to overcome the difficulties encountered in using a continuous approximation, as discussed above. However, the filled function proposed by Zhu contains an exponential term, which consequently makes it difficult to determine a point in a lower basin [26,27]. Since then, several types of discrete filled functions with improved theoretical properties have been proposed in [16,27,28,32,33,42–44] to enhance computational efficiency.

The discrete filled function approach can be described as follows. An initial point is chosen and a local search is applied to find an initial discrete local minimizer. Then, an auxiliary function, called a filled function, is constructed at this local minimizer. By minimizing the filled function, either an improved discrete local minimizer is found or the boundary of the feasible region is reached. The discrete local minimizer of the filled function usually becomes a new starting point for minimizing the original objective with the hope of finding an improved point compared to the first local minimizer. A new filled function is constructed at this improved point. The process is repeated until no improved local minimizer of the earlier filled function can be found. The final discrete local minimizer is then taken as an approximation of the global minimizer.

If a local minimizer of the filled function cannot be found after repeated searches terminate on the boundary of the box constrained feasible region, the parameters defining the filled functions are adjusted and the search is repeated. This adjustment of the parameters continues until the parameters reach their predetermined bounds; the best solution obtained so far is then taken as the global minimizer. Note that some filled functions have one parameter (such as those in [16,32,43]), while the rest are equipped with two parameters. The latter filled functions often have one parameter which is partially dependent on the other and this requires additional steps when tuning the parameters in order to satisfy the required convergence criteria. Note that each filled function discussed here has unique characteristics. The complexity of each filled function is also dependent on its associated algorithm, as discussed in detail in the following sections.

Consider the following nonlinear discrete optimization problem:

$$\min f(\mathbf{x}), \quad \text{s.t. } \mathbf{x} \in X, \tag{1}$$

where $X = \{\mathbf{x} \in \mathbb{Z}^n | \mathbf{x}_{i,min} \leqslant \mathbf{x}_i \leqslant \mathbf{x}_{i,max}\}$, $\mathbb{Z}^n$ is the set of integer points in $\mathbb{R}^n$, and $\mathbf{x}_{i,min}$, $\mathbf{x}_{i,max}$, $i = 1,\ldots,n$, are given bounds. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be any two distinct points in the box constrained set X and observe the following assumptions:

Assumption 1. There exists a constant $\mathcal{K}$ satisfying

$$1 \leqslant \max_{\substack{\mathbf{x}_1,\mathbf{x}_2 \in X \\ \mathbf{x}_1 \neq \mathbf{x}_2}} \|\mathbf{x}_1 - \mathbf{x}_2\| \leqslant \mathcal{K} < \infty,$$

where $\|\cdot\|$ is the Euclidean norm.

Assumption 2. There exists a constant $\mathcal{L}$, $0 < \mathcal{L} < \infty$, such that

$$|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leqslant \mathcal{L}\|\mathbf{x}_1 - \mathbf{x}_2\|.$$

Most discrete filled function methods are designed to solve box constrained problems. Unconstrained and more generally constrained problems may be converted into an equivalent box constrained form. For example, consider the following unconstrained discrete optimization problem,

$$\min f(\mathbf{x}), \quad \text{s.t. } \mathbf{x} \in \mathbb{Z}^n. \tag{2}$$

If f is coercive, i.e., $f(\mathbf{x}) \to +\infty$ as $\|\mathbf{x}\| \to +\infty$, then there exists a box which contains all discrete minimizers of f. Hence, the formulation in (2) can be transformed into an equivalent formulation in (1) and thus can be solved by any discrete filled function method. Many discrete filled function algorithms in the literature, such as those in [16,27,28,44], are also directly applicable to linearly constrained problems as long as the resulting feasible region is convex and pathwise connected.

As for generally constrained problems, the nonlinear constraints are usually handled with a penalty method. Consider the following general nonlinear constrained discrete optimization problem,

$$\min g_0(\mathbf{x}), \quad \text{s.t. } \mathbf{x} \in \Lambda, \tag{3}$$

where $\Lambda = \{\mathbf{x} \in \mathbb{Z}^n : g_i(\mathbf{x}) \leqslant 0,\ i = 1, \ldots, m\}$ and $\mathbb{Z}^n$ is the set of integer points in $\mathbb{R}^n$. In [26], the constrained problem (3) is converted into an equivalent box constrained problem by adding a penalty term to the objective function f, i.e.,

$$f(\mathbf{x}) = g_0(\mathbf{x}) + \alpha_0 \sum_{i=1}^{m} \max\{0, g_i(\mathbf{x})\} \tag{4}$$

or

$$f(\mathbf{x}) = g_0(\mathbf{x}) + \alpha_0 \sum_{i=1}^{m} [\max\{0, g_i(\mathbf{x})\}]^2, \tag{5}$$

where α_0 is a sufficiently large parameter. Note that it is difficult to determine an exact penalty parameter when solving these NP-hard problems and thus only approximate solutions can be determined. Note also that the discrete filled function method in [43] takes a different approach and incorporates constraints directly into the formulation of the filled function. The purpose of this paper is to review several discrete filled functions and their associated algorithms as proposed in the literature. The remainder of this paper is organized as follows. We review the basic discrete optimization concepts and present a generic discrete filled function algorithm in the following section. Then, we discuss several individual discrete filled function formulations, their properties, and particulars of their associated algorithms in Section 3. The performances of selected filled function algorithms when applied to several test problems are compared in Section 4.

2. Discrete optimization: concepts and approach

2.1. Preliminary concepts

We recall some definitions and concepts used in the discrete optimization area.

Definition 1. A sequence $\{\mathbf{x}^{(i)}\}_{i=0}^{k+1}$ between two distinct points $\mathbf{x}^*$ and $\mathbf{x}^{**}$ in X is a discrete path in X if $\mathbf{x}^{(0)} = \mathbf{x}^*$, $\mathbf{x}^{(k+1)} = \mathbf{x}^{**}$, $\mathbf{x}^{(i)} \in X$ for all i, $\mathbf{x}^{(i)} \neq \mathbf{x}^{(j)}$ for $i \neq j$, and $\|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\| = 1$ for all i. If such a discrete path exists, then $\mathbf{x}^*$ and $\mathbf{x}^{**}$ are pathwise connected in X. If every two distinct points in X are pathwise connected in X, then X is a pathwise connected set.

Definition 2. For any $\mathbf{x} \in X$, the *neighbourhood* of $\mathbf{x}$ is defined by $N(\mathbf{x}) = \{\mathbf{w} \in X | \mathbf{w} = \mathbf{x} \pm \mathbf{e}_i;\ i = 1, 2, \ldots, n\}$. Here, $\mathbf{e}_i$ denotes the ith standard unit basis vector of $\mathbb{R}^n$, with the ith component equal to one and all other components equal to zero.

Definition 3. The set of all feasible directions at $\mathbf{x} \in X$ is defined by $\mathcal{D}(\mathbf{x}) = \{\mathbf{d} \in \mathbb{R}^n : \mathbf{x} + \mathbf{d} \in N(\mathbf{x})\} \subset E = \{\pm\mathbf{e}_1, \ldots, \pm\mathbf{e}_n\}$.

Definition 4. $\mathbf{d} \in \mathcal{D}(\mathbf{x})$ is a descent direction of f at $\mathbf{x}$ if $f(\mathbf{x} + \mathbf{d}) < f(\mathbf{x})$.

Definition 5. $\mathbf{d}^* \in \mathcal{D}(\mathbf{x})$ is a discrete steepest descent direction of f at $\mathbf{x}$ if it is a descent direction and $f(\mathbf{x} + \mathbf{d}^*) \leqslant f(\mathbf{x} + \mathbf{d})$ for any $\mathbf{d} \in \mathcal{D}(\mathbf{x})$.

Definition 6. $\mathbf{x}^* \in X$ is a *local minimizer* of X if $f(\mathbf{x}^*) \leqslant f(\mathbf{x})$ for all $\mathbf{x} \in N(\mathbf{x}^*)$. If $f(\mathbf{x}^*) < f(\mathbf{x})$ for all $\mathbf{x} \in N(\mathbf{x}^*) \backslash \mathbf{x}^*$, then $\mathbf{x}^*$ is a *strict local minimizer* of f.

Definition 7. $\mathbf{x}^*$ is a *global minimizer* of f if $f(\mathbf{x}^*) \leqslant f(\mathbf{x})$ for all $\mathbf{x} \in X$. If $f(\mathbf{x}^*) < f(\mathbf{x})$ for all $\mathbf{x} \in X \backslash \mathbf{x}^*$, then $\mathbf{x}^*$ is a *strict global minimizer* of f.

Definition 8. $\mathbf{x}$ is a vertex of X if for each $\mathbf{d} \in \mathcal{D}(\mathbf{x})$, $\mathbf{x} + \mathbf{d} \in X$ and $\mathbf{x} - \mathbf{d} \notin X$. Let $\widetilde{X}$ denote the set of vertices of X.

Definition 9. $B^* \subset f$ is a discrete basin of f corresponding to $\mathbf{x}^*$ if it satisfies the following conditions:

- It is pathwise connected.
- It contains $\mathbf{x}^*$.
- For each $\mathbf{x} \in B^*$, any connected path consisting of descent steps and starting at $\mathbf{x}$ converges to $\mathbf{x}^*$.

Definition 10. Let $\mathbf{x}^*$ and $\mathbf{x}^{**}$ be two distinct local minimizers of f. If $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$, then the discrete basin B^{**} of f associated with $\mathbf{x}^{**}$ is said to be lower than the discrete basin B^* of f associated with $\mathbf{x}^*$

Definition 11. Let $\mathbf{x}^*$ be a local minimizer of $-f$. The discrete basin of $-f$ at $\mathbf{x}^*$ is called a discrete hill of f at $\mathbf{x}^*$.

Definition 12. Let $S_L = \{\mathbf{x} \in X : f(\mathbf{x}) < f(\mathbf{x}^*)\}$ and $S_U = \{\mathbf{x} \in X : f(\mathbf{x}) \geqslant f(\mathbf{x}^*)\}$.

A discrete filled function is defined as an auxiliary function constructed at a local minimizer of the original function f, where the local minimizer of f becomes a local maximizer of the auxiliary function. By minimizing the auxiliary function, the search moves away from the current local minimizer in the hope of escaping the basin associated with this minimizer. Note that the auxiliary function is defined in terms of one or more parameters and needs to possess certain properties, details of which are discussed in the next section.

2.2. Generic discrete filled function approach

We present the generic framework of a discrete filled function algorithm. The main algorithm requires repeated searches for a local minimum. We thus state the local search as a separate algorithm (see Algorithm 1). The global algorithm involves repeated construction of an auxiliary function in the hope of escaping basins associated with local minimizers.

Algorithm 1 (*Discrete Steepest Descent Method*).

1. Choose an initial point $\mathbf{x} \in X$.
2. If $\mathbf{x}$ is a local minimizer of f, then stop. Otherwise, find the discrete steepest descent direction $\mathbf{d}^* \in \mathcal{D}(\mathbf{x})$ of f.
3. Set $\mathbf{x} := \mathbf{x} + \mathbf{d}^*$. Go to Step 2.

Remark 1. Note that some methods in the literature, such as [32,46], merely require a discrete descent direction at Step 2, rather than a discrete steepest descent direction.

Algorithm 2 (*Discrete Filled Function Method*).

1. Initialization.
   Set the bounds of each parameter in the formulation of the discrete filled function.
   Initialize the parameters.
   Identify suitable reduction or increment strategies for each parameter. Choose an initial starting point $\mathbf{x}_0 \in X$.
2. Local search of the original function.
   Starting from $\mathbf{x}_0$, minimize $f(\mathbf{x})$ using Algorithm 1 to obtain a local minimizer $\mathbf{x}^*$ of f.
3. Neighbourhood search.
   (a) Identify the neighbourhood of $\mathbf{x}^*$ as $N(\mathbf{x}^*) = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q\}$, where q is the total number of points in $N(\mathbf{x}^*)$, $q \leqslant 2n$. Set $\ell = 1$.
   (b) Define the current point, $\mathbf{x}_c := \mathbf{w}_\ell$.
4. Local search of discrete filled function.
   Let $G_{\mathbf{x}^*}$ denote the discrete filled function associated with $\mathbf{x}^*$.
   Minimize $G_{\mathbf{x}^*}$ using Algorithm 1 starting from $\mathbf{x}_c$.
   Let $\acute{\mathbf{x}}$ be the obtained local minimizer of $G_{\mathbf{x}^*}$.
5. Checking the status of $\acute{\mathbf{x}}$.
   If $f(\acute{\mathbf{x}}) < f(\mathbf{x}^*)$, set $\mathbf{x}_0 := \acute{\mathbf{x}}$ and go to Step 2. Otherwise, go to Step 6.
6. Checking other search directions.
   At this point, the algorithms in [27,28,44] will adjust the parameters of the filled function and return to Step 4 if $\acute{\mathbf{x}} \in X \setminus \widetilde{X}$.
   Otherwise, along with most of the remaining algorithms, they set $\ell := \ell + 1$.
   If $\ell \leqslant q$, all of the algorithms return to Step 3(b).
   Otherwise, the parameters of the filled function are adjusted before returning to Step 3(a).
   If all the parameters of the filled function exceed their prescribed bounds anywhere in this step, the current value of $\mathbf{x}^*$ is taken as the global minimizer.

Remark 2. Some methods in the literature, such as [32,33,44], replace $N(\mathbf{x}^*)$ in Step 3 with $M = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q\}$, where $\mathbf{w}_i$, $i = 1, \ldots, q$, are randomly chosen from X. Also, q needs to be chosen by the user in this case.

Remark 3. Some algorithms [16,27,28,33,42] do not require a local minimizer of $G_{\mathbf{x}^*}$ in Step 4. Instead, in the attempt to reduce $G_{\mathbf{x}^*}$, if any point $\mathbf{x}_k$ is found such that $f(\mathbf{x}_k) < f(\mathbf{x}^*)$, they set $\mathbf{x}_0 := \mathbf{x}_k$ and go back to Step 2.

Remark 4. Both functions f and $G_{\mathbf{x}^*}$ are minimized subject to X, except in [43]. The feasible regions of f and $G_{\mathbf{x}^*}$ are defined by A and X, respectively, in [43].

Remark 5. Note that the methods in [16,27,28,44] define X via upper and lower bounds on the variables as well as a set of linear inequality constraints.

Remark 6. A slightly different approach is proposed in [27] for Step 4. If f and $G_{\mathbf{x}^*}$ share at least one common descent direction, the authors choose a steepest descent direction which results in the maximum reduction for $f + G_{\mathbf{x}^*}$. If such a direction does not exist, the method reverts to find a steepest descent direction for $G_{\mathbf{x}^*}$ only.

For a clearer picture on how the filled function algorithm works, we consider an illustrative example in the next subsection.

2.3. Illustrative example

$$\begin{aligned} &\min \quad f(\mathbf{x}) = 2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 - x_1x_2 + x_2^2, \\ &\text{s.t.} \quad x_i = \frac{y_i}{1000}, \quad -2000 \leqslant y_1 \leqslant 2000, \quad -1500 \leqslant y_2 \leqslant 1500, \quad y_1, y_2 \text{ integers.} \end{aligned} \tag{6}$$

Problem (6) is the 3-hump back camel function proposed in [5] which has 1.2007001×10^7 feasible points. This box constrained problem has a global minimum solution at $\mathbf{x}^*_{\text{global}} = [0, 0]^T$ with $f\left(\mathbf{x}^*_{\text{global}}\right) = 0$. The discrete filled function method in [27] is used to solve this problem. The algorithm begins with a point $\mathbf{x}_0 = [1.500, 1.500]^T$ with $f(\mathbf{x}_0) = 1.0828125$. By using the discrete steepest descent method, an initial local minimizer of $\mathbf{x}_1^* = [1.748, 0.874]^T$ is found with $f(\mathbf{x}_1^*) = 0.2986396$. Next, a discrete filled function, $G_{\mathbf{x}_1^*}$, is constructed at $\mathbf{x}_1^*$. Starting with a point in $N(\mathbf{x}_1^*)$, $\mathbf{x}_c = [1.749, 0.874]^T$, Algorithm 1 is used to minimize $G_{\mathbf{x}_1^*}$ and a local minimizer, $\acute{\mathbf{x}} = [0.302, 0.535]^T$, with $f(\acute{\mathbf{x}}) = 0.2984554$, is found. Since $f(\acute{\mathbf{x}}) < f(\mathbf{x}_1^*)$, the original function f is minimized once more, starting at $\mathbf{x}_0 = \acute{\mathbf{x}}$, and the second local minimizer $\mathbf{x}_2^* = [0, 0]^T$, with $f(\mathbf{x}_2^*) = 0$, is obtained. Next, a new discrete filled function $G_{\mathbf{x}_2^*}$ is constructed at $\mathbf{x}_2^* = [0, 0]^T$. A neighbourhood point of $\mathbf{x}_2^* = [0, 0]^T$, namely $\mathbf{x}_c = [1, 0]^T$, is chosen, and $G_{\mathbf{x}_2^*}$ is minimized starting at $\mathbf{x}_c$. The local minimizer of $G_{\mathbf{x}_2^*}$ is a vertex, $\acute{\mathbf{x}} = [2.000, 1.500]^T$, but $f(\acute{\mathbf{x}}) > f(\mathbf{x}_2^*)$. Other searches for a minimum of $G_{\mathbf{x}_2^*}$ in a lower basin are then carried out, starting from $[0, 1]^T$, $[-1, 0]^T$, and

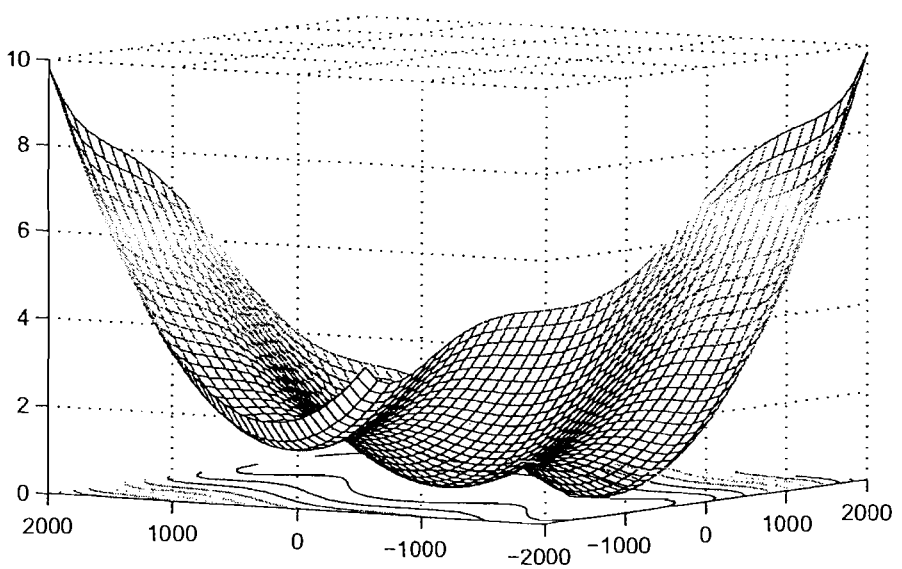

Fig. 1. The 3-hump back camel function.

$[0,-1]^T$, respectively. Since none of these yield an improved point, the parameter of $G_{\mathbf{x}_2^*}$ is adjusted. The revised $G_{\mathbf{x}_2^*}$ is then minimized once more starting from each of these neighbourhood points in turn. When no local minimizer of $G_{\mathbf{x}_2^*}$ in a lower basin is found and the termination criteria is met, $\mathbf{x}_2^* = [0,0]^T$ is taken to be the global solution (see Fig. 1).

In the next section, we discuss and analyze various discrete filled function methods from the literature.

3. Discrete filled function methods

3.1. Discrete filled function in Zhu [46]

Zhu is believed to be the first researcher to adapt the continuous filled function approach directly for solving discrete optimization problems. Let $\mathbf{x}^*$ denote the current discrete local minimizer. A filled function dependent on parameters θ and p is defined as

$$G_{\theta,p,\mathbf{x}^*}(\mathbf{x}) = \frac{1}{\theta + f(\mathbf{x})} \exp\left(\frac{\|\mathbf{x}-\mathbf{x}^*\|^2}{-p^2}\right). \tag{7}$$

Assuming that p and θ are chosen so that

$$0 < \theta + f(\mathbf{x}^*) < h$$

and

$$p^2 \ln\left(\frac{\theta+\bar{f}}{\theta+f(\mathbf{x}^*)}\right) < 1,$$

where $\bar{f}$ is an upper bound of f over X and $h \leqslant \min\{|f(\mathbf{x}_1) - f(\mathbf{x}_2)| : f(\mathbf{x}_1) \neq f(\mathbf{x}_2), \mathbf{x}_j \in X, j = 1,2\}$, the filled function (7) has the following properties:

- $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}^* + \mathbf{d}) < G_{\theta,p,\mathbf{x}^*}(\mathbf{x}^*)$, for all $\mathbf{d} \in \mathcal{D}(\mathbf{x}^*)$.
- Given $f(\mathbf{x}_1) \geqslant f(\mathbf{x}^*)$, $f(\mathbf{x}_2) \geqslant f(\mathbf{x}^*)$, and $\|\mathbf{x}_2 - \mathbf{x}^*\|^2 < \|\mathbf{x}_1 - \mathbf{x}^*\|^2$, $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1) < G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_2)$ (i.e. if f increases, $G_{\theta,p,\mathbf{x}^*}$ decreases).
- For any $\mathbf{x} \in X$,
  - $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}) < 0 \Longleftrightarrow f(\mathbf{x}) < f(\mathbf{x}^*)$;
  - $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}) > 0 \Longleftrightarrow f(\mathbf{x}) \geqslant f(\mathbf{x}^*)$.
- Let $\mathbf{x}_1 \in X$ such that $f(\mathbf{x}_1) \geqslant f(\mathbf{x}^*)$.
  - If there exists $\mathbf{d} \in \mathcal{D}(\mathbf{x}_1)$ such that $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1 + \mathbf{d}) < 0$; or
  - If $|\{\mathbf{d} \in \mathcal{D}(\mathbf{x}_1) : \mathbf{x}_1 + \mathbf{d} \in X\}| = n$ and there exists $\mathbf{d} \in \mathcal{D}(\mathbf{x}_1)$ such that $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1 + \mathbf{d}) < G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1)$; or
  - If $|\{\mathbf{d} \in \mathcal{D} : \mathbf{x}_1 + \mathbf{d} \in X\}| > n$; then there exists some $\mathbf{d} \in \mathcal{D}(\mathbf{x}_1)$ such that $G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1 + \mathbf{d}) < G_{\theta,p,\mathbf{x}^*}(\mathbf{x}_1) < G_{\theta,p,\mathbf{x}^*}(\mathbf{x}^*)$.
- Any discrete local minimizer of the discrete filled function $G_{\theta,p,\mathbf{x}^*}$ must be in the set S_L or $\widetilde{X}$.

Zhu suggests that the algorithm should stop when all searches for a minimum of $G_{\theta,p,\mathbf{x}^*}$ starting in $N(\mathbf{x}^*)$ terminate at vertices without finding an improved point of f. Note that the algorithm in [46] does not require updating of the parameters θ and p. Thus, the final $\mathbf{x}^*$ is assumed to be the global minimum. Two numerical examples are demonstrated to test the efficiency of this filled function. However, the disadvantage of his method is that it is almost impossible to find a negative filled function value that would indicate that a point in a lower basin exists. This is because the discrete filled function contains an exponential term, making it ill conditioned and also leading to poor efficiency as noted in [27]. In addition, it is difficult to determine suitable values of h and $\bar{f}$, thus making it difficult to find suitable values for parameters θ and p.

3.2. Discrete filled function in Ng et al. [28]

A new discrete filled function with improved theoretical properties was proposed in [28] several years later. Recall that B^* denotes a discrete basin of f that contains the current discrete local minimizer $\mathbf{x}^*$. According to [28], a function $G_{\mu,\rho,\mathbf{x}^*}$ is defined to be a discrete filled function of f at $\mathbf{x}^*$ if it satisfies the following:

- $\mathbf{x}^*$ is a strict local maximizer of $G_{\mu,\rho,\mathbf{x}^*}$.
- $G_{\mu,\rho,\mathbf{x}^*}$ has no discrete local minimizers in B^* or in any discrete basin of f higher than B^*.
- If f has a discrete basin B^{**} at $\mathbf{x}^{**}$ which is lower than B^*, then there is a discrete point $\acute{\mathbf{x}} \in B^{**}$ that minimizes $G_{\mu,\rho,\mathbf{x}^*}$ on a connected discrete path $\{\mathbf{x}^*, \ldots, \acute{\mathbf{x}}, \ldots, \mathbf{x}^{**}\}$ in X.

The discrete filled function proposed in [28] is

$$G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}) = f(\mathbf{x}^*) - \min[f(\mathbf{x}^*), f(\mathbf{x})] - \rho\|\mathbf{x}-\mathbf{x}^*\|^2 + \mu\{\max[0, f(\mathbf{x}) - f(\mathbf{x}^*)]\}^2, \tag{8}$$

where ρ and μ are parameters which satisfy certain properties as detailed below.

- Recall the meaning of $\mathcal{K}$ and $\mathcal{L}$ from Assumptions 1 and 2. Suppose that $\bar{\mathbf{x}} \in S_U$.
  - If $\rho > 0$ and $0 \leqslant \mu < \frac{\rho}{\mathcal{L}^2}$, then $G_{\mu,\rho,\mathbf{x}^*}(\bar{\mathbf{x}}) < 0 = G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*)$.
  - If $\rho > 0$ and $0 \leqslant \mu \leqslant \frac{\rho}{2\mathcal{K}^2\mathcal{L}^2}$, then for each $\bar{\mathbf{d}} \in \mathcal{D}(\bar{\mathbf{x}})$ such that $f(\bar{\mathbf{x}} + \bar{\mathbf{d}}) \geqslant f(\mathbf{x}^*)$ and $\|\bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^*\| > \|\bar{\mathbf{x}} - \mathbf{x}^*\|, G_{\mu,\rho,\mathbf{x}^*}(\bar{\mathbf{x}} + \bar{\mathbf{d}}) < G_{\mu,\rho,\mathbf{x}^*}(\bar{\mathbf{x}}) < 0 = G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*)$.
- If $\rho > 0$ and $0 \leqslant \mu < \frac{\rho}{\mathcal{L}^2}$, then $\mathbf{x}^*$ is a strict local maximizer of $G_{\mu,\rho,\mathbf{x}^*}$. If $\mathbf{x}^*$ is a global minimizer of f, then $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*) < 0$, for all $\mathbf{x} \in X \backslash \mathbf{x}^*$.
- Let $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}^*$ be three distinct points in X. If $\|\mathbf{x}_2 - \mathbf{x}^*\| > \|\mathbf{x}_1 - \mathbf{x}^*\|$, then $1 \leqslant \frac{\|\mathbf{x}_2 - \mathbf{x}_1\|}{\|\mathbf{x}_2 - \mathbf{x}^*\| - \|\mathbf{x}_1 - \mathbf{x}^*\|} < 2\mathcal{K}^2$.
- Let $\mathbf{x}_1, \mathbf{x}_2 \in X$ be two points such that $0 < \|\mathbf{x}_1 - \mathbf{x}^*\| < \|\mathbf{x}_2 - \mathbf{x}^*\|$ and $f(\mathbf{x}^*) \leqslant f(\mathbf{x}_1) \leqslant f(\mathbf{x}_2)$. If $\rho > 0$ and $0 \leqslant \mu \leqslant \frac{\rho}{2\mathcal{K}^2\mathcal{L}^2}$, then
  - $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_2) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_1) < 0 = G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*)$;
  - $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*)$ has no local minimizers in B^* or in any discrete basin of f higher than B^*.
- For every $\acute{\mathbf{x}}, \mathbf{x}^* \in X$, there exists $\mathbf{d} \in E$ such that $\|\acute{\mathbf{x}} + \mathbf{d} - \mathbf{x}^*\| > \|\acute{\mathbf{x}} - \mathbf{x}^*\|$.
- Let $\mathbf{x}^* \in X$ and $\acute{\mathbf{x}} \in X$ be the local minimizers of f and $G_{\mu,\rho,\mathbf{x}^*}$, respectively. If $\rho > 0$ and $0 \leqslant \mu \leqslant \frac{\rho}{2\mathcal{K}^2\mathcal{L}^2}$, then
  - $f(\acute{\mathbf{x}} + \acute{\mathbf{d}}) < f(\mathbf{x}^*)$ for all $\acute{\mathbf{d}} \in \mathcal{D}(\acute{\mathbf{x}})$ when $f(\acute{\mathbf{x}}) \geqslant f(\mathbf{x}^*)$.
  - $\acute{\mathbf{x}}$ is in a basin B^{**} (associated with a local minimum $\mathbf{x}^{**}$) of f which is lower than basin B^* (associated with $\mathbf{x}^*$).

Both μ and ρ are initialized as 1. This filled function ensures that a local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ is either a better point in a lower basin or a vertex of X. It is not necessary to find the minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ if a point $\mathbf{x}_k$ with $f(\mathbf{x}_k) < f(\mathbf{x}^*)$ is found in Step 4 of Algorithm 2. Since $\mathbf{x}_k$ is an improved point, the algorithm sets $\mathbf{x}_0 := \mathbf{x}_k$ and returns to Step 2 to minimize the original function f. If the minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ is not a vertex, μ is reduced via $\mu := \mu/10$ and $G_{\mu,\rho,\mathbf{x}^*}$ is minimized once more starting at the same $\mathbf{x}_c$. When no improved point is found after the minimization process for $G_{\mu,\rho,\mathbf{x}^*}$ ends up at a vertex, then set $\ell := \ell + 1$ and return to Step 3(b). If $\ell > q$, ρ is reduced. The algorithm terminates when the lower bound of ρ, ρ_L, is met. Several test problems were investigated in [28] and the proposed discrete filled function was shown to be efficient in solving large scale problems involving up to 200 variables. Note that ρ_L was set to 1 for the computations in [28] and further reduction of ρ was not necessary since all test problems yielded the global solution when $\rho = 1$. According to one of the characteristics of this filled function, $\acute{\mathbf{x}}$, the local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$, lies on a discrete path $\{\mathbf{x}^*, \ldots, \acute{\mathbf{x}}, \ldots, \mathbf{x}^{**}\}$ in X that connects the current basin B^* at $\mathbf{x}^*$ to a lower basin B^{**}. However, the properties of this filled function do not guarantee that $\acute{\mathbf{x}}$ is a true minimizer of the original function. A revised discrete filled function is proposed in [27] to overcome this difficulty.

3.3. Discrete filled function in Ng et al. [27]

Based on the work in [28], a new discrete filled function $G_{\mu,\rho,\mathbf{x}^*}$ at $\mathbf{x}^*$ is defined as follows:

$$G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}) = A_\mu(f(\mathbf{x}) - f(\mathbf{x}^*)) - \rho\|\mathbf{x} - \mathbf{x}^*\|, \tag{9}$$

$$A_\mu(y) = y \cdot \mu\left[(1-c)\left(\frac{1-c\mu}{\mu - c\mu}\right)^{-y/\omega} + c\right],$$

where $\omega > 0$ is a sufficiently small number and $0 < c \leqslant 1$ is a constant. The function $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x})$ is a discrete filled function when certain conditions of the parameters μ and ρ are satisfied as detailed in the following conditions:

- $\mathbf{x}^*$ is a strict local maximizer of $G_{\mu,\rho,\mathbf{x}^*}$.
- $G_{\mu,\rho,\mathbf{x}^*}$ has no local minimizer in the set $S_U \setminus \bar{X}$.
- $\mathbf{x}^{**} \in X \setminus \bar{X}$ is a local minimizer of f if and only if $\mathbf{x}^{**}$ is a local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$. In short, $\mathbf{x}^{**} \in S_L$.
- If $\rho > 0$ and $0 < \mu < \min\{1, \frac{\rho}{\mathcal{L}}\}$, then $\mathbf{x}^*$ is a strict local maximizer of $G_{\mu,\rho,\mathbf{x}^*}$. If $\mathbf{x}^*$ is a global minimizer of f, then $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}) < 0$ for all $\mathbf{x} \in X \backslash \mathbf{x}^*$.
- Let $\bar{\mathbf{d}} \in \mathcal{D}(\bar{\mathbf{x}})$ be a feasible direction at $\bar{\mathbf{x}} \in S_U$ such that $\|\bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^*\| > \|\bar{\mathbf{x}} - \mathbf{x}^*\|$. If $\rho > 0$ and $0 < \mu < \min\left\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\right\}$, then $G_{\mu,\rho,\mathbf{x}^*}(\bar{\mathbf{x}} + \bar{\mathbf{d}}) < G_{\mu,\rho,\mathbf{x}^*}(\bar{\mathbf{x}}) < 0 = G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}^*)$.
- Let $\mathbf{x}^{**}$ be a strict local minimizer of f with $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$. If $\rho > 0$ is sufficiently small and $0 < \mu < 1$, then $\mathbf{x}^{**}$ is a strict local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$.
- Let $\acute{\mathbf{x}}$ be a strict local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ and $\bar{\mathbf{d}} \in \mathcal{D}(\acute{\mathbf{x}})$ be a feasible direction at $\acute{\mathbf{x}}$ such that $\|\acute{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^*\| > \|\acute{\mathbf{x}} - \mathbf{x}^*\|$. If $\rho > 0$ is sufficiently small and $0 < \mu < \min\left\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\right\}$, then $\acute{\mathbf{x}}$ is a local minimizer of f.
- Assume that every local minimizer of f is strict. Suppose that $\rho > 0$ is sufficiently small and $0 < \mu < \min\left\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\right\}$. Then, $\mathbf{x}^{**} \in X \setminus \bar{X}$ is a local minimizer of f with $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$ if and only if $\mathbf{x}^{**}$ is a local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$.

This is an improved version of the discrete filled function in [28], to ensure $\acute{\mathbf{x}}$ coincides with $\mathbf{x}^{**}$. In other words, every local minimizer of the discrete filled function $G_{\mu,\rho,\mathbf{x}^*}$ is also a local minimizer for the original function f. Both μ and ρ are initialized as 0.1. The parameter μ is reduced if $\acute{\mathbf{x}}$ is neither a vertex nor an improved point by setting $\mu := \mu/10$ and returning to Step 3(a). When all the searches end up at vertices, set $\ell := \ell + 1$ and return to Step 3(b). Another parameter ρ is adjusted when $\ell > q$. Similar to [28], the algorithm for minimizing $G_{\mu,\rho,\mathbf{x}^*}$ exits prematurely when an improved point $\mathbf{x}_k$ with $f(\mathbf{x}_k) < f(\mathbf{x}^*)$ is found in Step 4 of Algorithm 2. The algorithm sets $\mathbf{x}_0 := \mathbf{x}_k$ and returns to Step 2 to minimize the original

function f. Note that a direction which yields the greatest improvement of $f + G_{\mu,\rho,\mathbf{x}^*}$ is chosen when minimizing $G_{\mu,\rho,\mathbf{x}^*}$, assuming that a direction for improving f and $G_{\mu,\rho,\mathbf{x}^*}$ does exist simultaneously. If such a direction does not exist, the algorithm chooses the steepest descent direction such that $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d}^*) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c)$. The algorithm terminates when $\rho_L = 0.1$. Note that ρ is fixed at 0.1, since all test problems yield a global solution with this setting. The filled function in (9) is shown to increase computational efficiency when compared with that in [28]. Several test problems with up to 1.38×10^{104} feasible points were solved using this method.

3.4. Discrete filled function in Yang and Liang [42]

A two parameter exponential filled function,

$$G_{a,b,\mathbf{x}^*}(\mathbf{x}) = \frac{1}{a + \|\mathbf{x} - \mathbf{x}^*\|}\Upsilon(\max\{f(\mathbf{x}) - f(\mathbf{x}^*) + b, 0\}), \tag{10}$$

where

$$\Upsilon(y) = \begin{cases} \exp(-a/y), & \text{if } y \neq 0, \\ 0, & \text{if } y = 0, \end{cases}$$

is introduced in [42]. Let S_M represent the set of discrete local minimizers of f, $a < 0$, and

$$0 < b < \max_{\substack{\mathbf{x}',\mathbf{x}'' \in S_M \\ f(\mathbf{x}'') < f(\mathbf{x}')}} (f(\mathbf{x}') - f(\mathbf{x}'')).$$

$G_{a,b,\mathbf{x}^*}$ is a discrete filled function of f if $G_{a,b,\mathbf{x}^*}(\mathbf{x})$ has the following properties:

- $\mathbf{x}^*$ is a strict discrete local maximizer of $G_{a,b,\mathbf{x}^*}$.
- $G_{a,b,\mathbf{x}^*}$ has no discrete local minimizers in S_U.
- If $\mathbf{x}^*$ is not a discrete global minimizer of f, then $G_{a,b,\mathbf{x}^*}$ does have a discrete minimizer $\hat{\mathbf{x}} \in S_L$.
- For any $\mathbf{x}, \mathbf{x}^* \in X$, there exists $\mathbf{d} \in \mathcal{D}(\mathbf{x})$ such that $\|\mathbf{x} + \mathbf{d} - \mathbf{x}^*\| < \|\mathbf{x} - \mathbf{x}^*\|$.
- Let $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}^*$ be three distinct points in X. If $\|\mathbf{x}_2 - \mathbf{x}^*\| > \|\mathbf{x}_1 - \mathbf{x}^*\|$, then $\frac{\|\mathbf{x}_1 - \mathbf{x}^*\|}{\|\mathbf{x}_2 - \mathbf{x}^*\|} < 1 - \frac{1}{2K^2}$.
- If, for any $\mathbf{x}_1, \mathbf{x}_2 \in X$,
  - $\|\mathbf{x}_2 - \mathbf{x}^*\| > \|\mathbf{x}_1 - \mathbf{x}^*\|$,
  - $f(\mathbf{x}_1) \geqslant f(\mathbf{x}^*)$,
  - $f(\mathbf{x}_2) - f(\mathbf{x}^*) + b > 0$,

  then $G_{a,b,\mathbf{x}^*}(\mathbf{x}_2) < G_{a,b,\mathbf{x}^*}(\mathbf{x}_1)$.

The parameters a and b are initialized as 0.01 and 1, respectively. When all the search directions from $\mathbf{x}^*$ have been utilized but no improved point of f is found (i.e., $\ell > q$) in Step 6 of Algorithm 2, the user proceeds as follows. If $a > 10^{-7}$, only a is reduced by a factor of 10. Otherwise, both a and b are reduced by a factor of 10. The algorithm terminates when $b \leqslant 10^{-5}$. Note that it is not necessary to find the minimizer of $G_{a,b,\mathbf{x}^*}$ for this algorithm. As long as a point $\mathbf{x}_k$ with $f(\mathbf{x}_k) < f(\mathbf{x}^*)$ is found when minimizing $G_{a,b,\mathbf{x}^*}$, the algorithm reverts to f to minimize the original function. As in [28], a local minimizer of this filled function is not guaranteed to be a true local minimizer of the original function f.

3.5. Discrete filled function in Shang and Zhang [32]

A third exponential filled function is suggested in [32]. Let $\mathbf{x}^*$ be the current local minimizer and choose any $\mathbf{x}_0$ such that $f(\mathbf{x}_0) \geqslant f(\mathbf{x}^*)$. According to [32], $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ is called a discrete filled function of f at $\mathbf{x}^*$ if $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ has the following properties:

- $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ has no local minimizer in $S_U \backslash \{\mathbf{x}_0\}$ and $\mathbf{x}_0$ is not necessarily a local minimizer of $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$.
- If $\mathbf{x}^*$ is not a global minimizer of f, there exists a local minimizer $\hat{\mathbf{x}} \in S_L$ of $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ such that $f(\hat{\mathbf{x}}) < f(\mathbf{x}^*)$.
- For any $\mathbf{x} \in X$, if $\mathbf{x} \neq \mathbf{x}_0$, there exists $\mathbf{d} \in \mathcal{D}(\mathbf{x})$ such that $\|\mathbf{x} + \mathbf{d} - \mathbf{x}_0\| < \|\mathbf{x} - \mathbf{x}_0\|$.

A discrete filled function is defined as

$$G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}(\mathbf{x}) = \zeta(\|\mathbf{x} - \mathbf{x}_0\|) - \xi(\varpi(1 - \exp(-[\min\{f(\mathbf{x}) - f(\mathbf{x}'), 0\}]^2))), \tag{11}$$

where $\varpi > 0$ is a parameter to be chosen and the prefixed point $\mathbf{x}_0$ satisfies $f(\mathbf{x}_0) \geqslant f(\mathbf{x}^*)$. The functions $\zeta(t)$ and $\xi(t)$ have the following characteristics:

- $\zeta(t)$ and $\xi(t)$ are strictly increasing for any $t \in [0, +\infty)$.
- $\zeta(0) = 0$ and $\xi(0) = 0$.
- $\xi(t) \to C > 0$ as $\mathbf{x} \to +\infty$, where $C \geqslant \max_{\mathbf{x} \in X} \zeta(\|\mathbf{x} - \mathbf{x}_0\|)$.

In addition, the following conditions hold for $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$:

- $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ has no local minimizer in $S_U\backslash\{\mathbf{x}_0\}$ for any $\varpi > 0$.
- Suppose $S_L \neq \emptyset$. If ϖ satisfies $\varpi > \frac{\xi^{-1}(C)\exp([f(\bar{\mathbf{x}}^*)-f(\mathbf{x}^*)]^2)}{\exp([f(\bar{\mathbf{x}}^*)-f(\mathbf{x}^*)]^2)-1}$, where $\bar{\mathbf{x}}^*$ is a global minimizer of f, then $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ has a local minimizer in S_L.
- Suppose that ε is a small positive constant and ϖ satisfies $\varpi > \frac{\xi^{-1}(C)\exp(\varepsilon^2)}{\exp(\varepsilon^2)-1}$. Then, given any $\mathbf{x}^*$ of f such that $f(\mathbf{x}^*) \geqslant f(\bar{\mathbf{x}}^*) + \varepsilon$, where $\bar{\mathbf{x}}^*$ is a global minimizer of f, $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ has at least one local minimizer in S_L.

Instead of performing a neighbourhood search in Step 3 of Algorithm 2, the implementation in [32] uses any initial point on the boundary of X to minimize $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$. In [32], the parameter ϖ is fixed to $400.5(10\sqrt{n}+1)$, where n is the dimension of a problem. For each subsequent initial point drawn from the boundary of X, $i := i + 1$, the algorithm terminates when $i = 10^n$. Every local minimizer of $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ is assumed to be an improved point (Step 5 of Algorithm 2 is bypassed). Though this filled function has only one fixed parameter, the local search of $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ can become computationally intensive due to the large number of initial points that may need to be tested before the termination criteria is met. A nonlinear box constrained problem with up to 1.71×10^5 feasible points was solved in [32]. Similar to [28,42], a local minimizer of the filled function $G_{\varpi,\mathbf{x}_0,\mathbf{x}^*}$ is not necessarily a local minimizer of the original function f. Furthermore, a prefixed point $\mathbf{x}_0$ is required at the beginning of the algorithm, resulting in the minimization process typically converging to $\mathbf{x}_0$ rather than an improved point of the original function. A refined formulation of this filled function is suggested in [33].

3.6. Discrete filled function in Shang and Zhang [33]

Let

$$G_{\delta,q,\mathbf{x}^*}(\mathbf{x}) = \frac{\ln(1 + q\max(f(\mathbf{x}) - f(\mathbf{x}^*) + \delta, 0))}{1 + \|\mathbf{x} - \mathbf{x}^*\|} \tag{12}$$

be a discrete filled function of f with $q > 0$ and

$$0 < \delta < \min_{\substack{\mathbf{x}_1,\mathbf{x}_2 \in X \\ \mathbf{x}_1 \neq \mathbf{x}_2}} |f(\mathbf{x}_1) - f(\mathbf{x}_2)|.$$

It has the following properties:

- $\mathbf{x}^*$ is a strict local maximizer of $G_{\delta,q,\mathbf{x}^*}$.
- If $f(\mathbf{x}) \geqslant f(\mathbf{x}^*)$ and $\mathbf{x} \neq \mathbf{x}^*$, then $\mathbf{x}$ is not a local minimizer of $G_{\delta,q,\mathbf{x}^*}$.
- If $\mathbf{x}^*$ is not a global minimizer of $f(\mathbf{x})$, there exists a local minimizer $\acute{\mathbf{x}}$ of $G_{\delta,q,\mathbf{x}^*}$ in S_L.
- If $\mathbf{x}_1, \mathbf{x}_2 \in X$ are two distinct points which satisfy
  - $f(\mathbf{x}_1) \geqslant f(\mathbf{x}^*)$ and $f\,\mathbf{x}_2) \geqslant f(\mathbf{x}^*)$, and
  - $\|\mathbf{x}_1 - \mathbf{x}^*\| > \|\mathbf{x}_2 - \mathbf{x}^*\| > 0$,

  then $G_{\delta,q,\mathbf{x}^*}(\mathbf{x}_1) < G_{\delta,q,\mathbf{x}^*}(\mathbf{x}_2)$ when $q > 0$ is sufficiently large.
- For any $\mathbf{x}_1, \mathbf{x}_2 \in X$ which satisfy
  - $f(\mathbf{x}_2) \geqslant f(\mathbf{x}^*) > f(\mathbf{x}_1)$, and
  - $\|\mathbf{x}_1 - \mathbf{x}^*\| > \|\mathbf{x}_2 - \mathbf{x}^*\| > 0$,

  $G_{\delta,q,\mathbf{x}^*}(\mathbf{x}_1) < G_{\delta,q,\mathbf{x}^*}(\mathbf{x}_2)$.

This filled function overcomes the prefixed point issue in [32] to ensure a better point of the original function is attained and suggests the use of an additional parameter. The initial settings for δ and q are $\delta_0 = 1$ and $q_0 = 100$, respectively. A random initial point in X is used to minimize $G_{\delta,q,\mathbf{x}^*}$ instead of a neighbourhood point, as suggested in Step 3 of Algorithm 2. If no local minimizer of $G_{\delta,q,\mathbf{x}^*}$ is found along the search from this random point, another initial point in X is drawn and $i := i + 1$. When $i > 2n$, $q := 10q$ and $q < 10^5$, the user sets $\delta := \delta/10$ and $q := q_0$ in Step 6 of Algorithm 2. Then, i is reset to 1 and $G_{\delta,q,\mathbf{x}^*}$ is minimized again from the same starting point with the new parameter values. Similar to the approach in [42], it is not necessary to find a minimizer of $G_{\delta,q,\mathbf{x}^*}$. The algorithm terminates when $\delta < 10^{-5}$ and $\ell = 2n$, where n refers to the dimension of the problem. Two test problems with up to 1.1739×10^{52} feasible points were solved in [33]. Since a local minimizer of this filled function is not necessarily a local minimizer of the original function f, further computation is needed to find the local minimizer of f in a lower basin for each local minimizer of $G_{\delta,q,\mathbf{x}^*}(\mathbf{x})$ found.

3.7. Discrete filled function in Yang and Zhang [44]

Suppose $\varphi(t)$ is a continuously differentiable function satisfying the following conditions:

- $\varphi(t) = \vartheta$ when $t \geqslant \epsilon$; $\varphi(t) = -\vartheta$ when $t \leqslant -\epsilon$.
- $\dot{\varphi}(t) \geqslant 0$, $-\epsilon \leqslant t \leqslant \epsilon$.
- $\varphi(0) = 0$.

Suppose also that a function $\eta(t)$ satisfies $\eta(0) = 0$ and $\dot{\eta}(t) > 0$, for $t \geqslant 0$. The filled function in [44] is given by

$$G_{\epsilon,\nu,\mathbf{x}^*}(\mathbf{x}) = \eta(\|\mathbf{x} - \mathbf{x}_0\|)\varphi(f(\mathbf{x}) - f(\mathbf{x}^*) + \nu), \tag{13}$$

where $\mathbf{x}_0$ is an arbitrary point in X, ϑ is a positive constant, and both ϵ and ν are problem-dependent parameters. The properties for this discrete filled function are as follows:

- The function $G_{\epsilon,\nu,\mathbf{x}^*}$ has no discrete local minimizer except at $\mathbf{x}_0$ in the region $S_1 = \{\mathbf{x} \in X : f(\mathbf{x}) \geqslant f(\mathbf{x}^*) + \epsilon - \nu\}$, where $\epsilon \geqslant \nu$.
- If $\nu = 0$, $G_{\epsilon,\nu,\mathbf{x}^*}(\mathbf{x})$ has no discrete local minimizer except at $\mathbf{x}_0$ in $S_2 = \{\mathbf{x} \in X : f(\mathbf{x}) \geqslant f(\mathbf{x}^*) + \epsilon\}$.
- If $\nu = \epsilon$, $G_{\epsilon,\nu,\mathbf{x}^*}(\mathbf{x})$ has no discrete local minimizer except at $\mathbf{x}_0$ in S_U.
- Given $\nu = 0$ or $\nu = \epsilon$, if ϵ is sufficiently small and $\mathbf{x}^*$ is not a discrete global minimizer of f, then $G_{\epsilon,\nu,\mathbf{x}^*}(\mathbf{x})$ does have a discrete local minimizer $\acute{\mathbf{x}}$ in S_L.
- If $\mathbf{x}^*$ is a global minimizer of f, then $\mathbf{x}_0$ is the unique discrete global minimizer of $G_{\epsilon,\nu,\mathbf{x}^*}(\mathbf{x})$ with $\nu > 0$.

The functions $\eta(t)$ and $\varphi(t)$ in (13) must be chosen carefully to ensure computational reliability and efficiency. As a guide, polynomial functions are suggested in [44] for both $\eta(t)$ and $\varphi(t)$. Based on the characteristics of this filled function, ϵ and ν are initialized as 1.0 and 0, respectively, so that there exists a local minimizer of $G_{\epsilon,\nu,\mathbf{x}^*}$ in a lower basin. The disadvantage of this filled function is that it depends heavily on the initial point $\mathbf{x}_0$ in computing $G_{\epsilon,\nu,\mathbf{x}^*}$. Thus, $\mathbf{x}_0$ has to be chosen carefully and plays a crucial role in finding a local minimizer of $G_{\epsilon,\nu,\mathbf{x}^*}$ such that $f(\bar{\mathbf{x}}^*) \leqslant f(\mathbf{x}_1^*) + \epsilon$, where $\bar{\mathbf{x}}^*$ is the global minimum of the original function. If a local minimizer of the filled function in a lower basin cannot be determined, then $\mathbf{x}_0$ is taken as its local minimizer, with suitable values of ϵ and ν, or $\mathbf{x}_0$ is assumed to be the global solution of the original function, which is not likely to happen in practice. The algorithm terminates when $\epsilon < 0.0001$.

3.8. Discrete filled function in Gu and Wu [16]

Gu and Wu propose the discrete filled function

$$G_{\varrho,\mathbf{x}^*}(\mathbf{x}) = \frac{1}{\|\mathbf{x} - \mathbf{x}^*\|^2 + 1} E_\varrho(f(\mathbf{x}) - f(\mathbf{x}^*)) + F_\varrho(f(\mathbf{x}) - f(\mathbf{x}^*)), \tag{14}$$

where

$$E_\varrho(y) = \begin{cases} 0, & y \leqslant -\varrho, \\ -\frac{2y^3}{\varrho^3} - \frac{3y^2}{\varrho^2} + 1, & -\varrho < y \leqslant 0, \\ 1, & y > 0, \end{cases}$$

and

$$F_\varrho(y) = \begin{cases} y + \varrho, & y \leqslant -\varrho, \\ \frac{(\varrho-2)y^3}{\varrho^3} + \frac{(\varrho-3)y^2}{\varrho^2} + 1, & -\varrho < y \leqslant 0, \\ 1, & y > 0. \end{cases}$$

Define $\beta_0 = \min_{\mathbf{x} \in S_L}(f(\mathbf{x}^*) - f(\mathbf{x}))$. If the function parameter ϱ satisfies

$$0 < \varrho \leqslant \beta_0,$$

then the following results hold:

- For all $\mathbf{x} \in X$, $f(\mathbf{x}) \geqslant f(\mathbf{x}^*)$ is equivalent to $G_{\varrho,\mathbf{x}^*}(\mathbf{x}) > 1$.
- $\mathbf{x}^*$ is not a global minimizer of f if and only if $S_L \neq \emptyset$ and $\beta_0 > 0$.
- $\mathbf{x} \in S_L$ is equivalent to $G_{\varrho,\mathbf{x}^*}(\mathbf{x}) \leqslant 0$.
- $\mathbf{x}^*$ is a strict discrete local maximizer of $G_{\varrho,\mathbf{x}^*}$.
- If $\mathbf{x}^*$ is not a global minimizer of f, then there exists a discrete local minimizer of $G_{\varrho,\mathbf{x}^*}$, denoted by $\acute{\mathbf{x}}$.
- $\acute{\mathbf{x}}$ is either in S_L or $\widetilde{X}$.
- Given $\mathbf{x}_1, \mathbf{x}_2 \in S_U$, $G_{\varrho,\mathbf{x}^*}(\mathbf{x}_1) > G_{\varrho,\mathbf{x}^*}(\mathbf{x}_2)$ is equivalent to $\|\mathbf{x}_1 - \mathbf{x}^*\| < \|\mathbf{x}_2 - \mathbf{x}^*\|$.

The parameter ϱ is initialized as 1. It is updated in Step 6 of Algorithm 2 by setting $\varrho := \varrho/10$ when all searches at $\mathbf{x}^*$ have been used (i.e. $\ell > q$) but no improved point of f is found. The algorithm terminates when $\varrho = 10^{-5}$. The one-parameter filled function suggested here guarantees that the minimizer of $G_{\varrho,\mathbf{x}^*}$ is also a minimizer of f. Based on this approach, a refined algorithm which is capable of dealing directly with nonlinear constraints is proposed in [43].

3.9. Discrete filled function in Yang et al. [43]

An extended study of the filled function method in [16] is given in [43] to deal with the nonlinear constrained problem (3). A one-parameter discrete filled function is defined as

$$G_{r,\mathbf{x}^*}(\mathbf{x}) = \left(\frac{1}{\|\mathbf{x}-\mathbf{x}^*\|^2+1}+1\right)\Gamma\left(H_r(f(\mathbf{x})-f(\mathbf{x}^*))+\sum_{i=1}^{m}H_r(g_i(\mathbf{x})-r)\right), \tag{15}$$

where

$$H_r(y)=\begin{cases}0, & y\leqslant -r,\\ \frac{(r-2)y^3}{r^3}+\frac{(2r-3)y^2}{r^2}+y+1, & -r<y\leqslant 0,\\ y+1, & y>0,\end{cases}$$

and

$$\Gamma(y)=\begin{cases}0, & y\leqslant 0.5,\\ -16y^3+36y^2-24y+5, & 0.5<y\leqslant 1,\\ 1, & y>1.\end{cases}$$

Let $\tilde{\beta}=\min\{\beta_0,\beta_1\}$, where

$$\beta_0=\min_{x\in S_L}(f(\mathbf{x}^*)-f(\mathbf{x}))$$

and

$$\beta_1=\min_{x\in X\setminus A}\max_{i\in\{1,\ldots,m\}}g_i(\mathbf{x}).$$

If the parameter r satisfies

$$0<r\leqslant\tilde{\beta},$$

$G_{r,\mathbf{x}^*}$ is said to be a discrete filled function at $\mathbf{x}^*$ and the following properties hold:

- $\mathbf{x}^*$ is a strict discrete local maximizer of $G_{r,\mathbf{x}^*}$ on X.
- If $\mathbf{x}^*$ is not a global minimizer of f, then there exists a $\hat{\mathbf{x}}\in S_L$ such that $\hat{\mathbf{x}}$ is a discrete local minimizer of $G_{r,\mathbf{x}^*}$.
- Any discrete local minimizer of $G_{r,\mathbf{x}^*}$ is either in S_L or in $\bar{X}$.
- Given $\mathbf{x}_1,\mathbf{x}_2\in X\setminus S_L$, $G_{r,\mathbf{x}^*}(\mathbf{x}_1)>G_{r,\mathbf{x}^*}(\mathbf{x}_2)$ if and only if $\|\mathbf{x}_1-\mathbf{x}^*\|<\|\mathbf{x}_2-\mathbf{x}^*\|$.
- $\mathbf{x}\in X\setminus S_L$ if and only if $G_{r,\mathbf{x}^*}(\mathbf{x})>1$.
- $\mathbf{x}\in S_L$ if and only if $G_{r,\mathbf{x}^*}(\mathbf{x})=0$.

Unlike the other filled functions discussed earlier, this filled function is capable of solving constrained nonlinear problems directly. Sets A and X are the feasible regions of f and $G_{r,\mathbf{x}^*}$, respectively. Note that as stated in [43], the algorithm is incomplete without justifying how to handle the non-feasibility issue of $\mathbf{x}_0$ if $\mathbf{x}_0\in X\setminus A$ happens to be used at the beginning of the algorithm. Based on correspondence with the main author in [43], we suggest an additional preliminary step before Step 1 in Algorithm 2 to check if $\mathbf{x}_0\in A$ before minimizing f. If this condition is satisfied, then follow Step 1 in Algorithm 2. Otherwise, set $\mathbf{x}^*:=\mathbf{x}_0$ and go directly to Step 3 in Algorithm 2. Since the local minimizer of the discrete filled function has to be tested for feasibility with respect to the original function, it is not guaranteed to be a local minimizer of f. Thus, further computation is needed for this single-parameter filled function approach for each minimizer of the filled function found. The parameter r is set as 1 at the beginning of the algorithm, reduced by $r:=r/10$ when $\ell>q$ in Step 6 of Algorithm 2, and the algorithm terminates when $r=10^{-5}$.

4. Solutions of test problems

In this section, we select several promising discrete filled function methods from those described in the previous section, based on their theoretical properties and algorithms. These functions are tested on several benchmark problems: Colville's function, Goldstein and Price's function, Beale's singular function, Powell's singular function, and Rosenbrock's function. Note that our aim is to simply compare the efficiency of different discrete filled function methods without necessarily solving high-dimensional problems. Note, though, that these methods have been demonstrated to solve problems involving up to 200 variables [27,28]. These algorithms are as follows:

- Algorithm A extracted from [28].
- Algorithm B extracted from [27].
- Algorithm C extracted from [42].
- Algorithm D extracted from [43].

The performance of each of the filled function method used in solving the test problem is summarized in the following subsections. The final optimal solution found for each algorithm is recorded by $\mathbf{x}^*_{\text{final}}$ with its corresponding objective value $f(\mathbf{x}^*_{\text{final}})$. The total number of original function evaluations, the total number of discrete filled function evaluations, and the ratio of the average number of original function evaluations to reach the global solution to the total number of feasible points are represented in Tables 1–5 by E_f, E_G, and R_E, respectively.

Table 1
Numerical results of Problem 1.

Type	$\mathbf{x}_0$	$\mathbf{x}^*_{\text{final}}$	$f(\mathbf{x}^*_{\text{final}})$	E_f	E_G	R_E
Algorithm A	$[1,1,0,0]^T$	$[1,1,1,1]^T$	0	2095	7058	0.010772261
	$[1,1,1,1]^T$	$[1,1,1,1]^T$	0	2086	7037	0.010725984
	$[-10,10,-10,10]^T$	$[1,1,1,1]^T$	0	3940	10,603	0.020259048
	$[-10,-5,0,5]^T$	$[1,1,1,1]^T$	0	2192	7056	0.011271024
	$[-10,0,0,-10]^T$	$[1,1,1,1]^T$	0	2226	7059	0.011445848
	$[0,0,0,0]^T$	$[1,1,1,1]^T$	0	2102	7060	0.010808254
Algorithm B	$[1,1,0,0]^T$	$[1,1,1,1]^T$	0	1426	5097	0.007332336
	$[1,1,1,1]^T$	$[1,1,1,1]^T$	0	1422	5076	0.007311768
	$[-10,10,-10,10]^T$	$[1,1,1,1]^T$	0	2674	5979	0.013749415
	$[-10,-5,0,5]^T$	$[1,1,1,1]^T$	0	1567	5134	0.008057342
	$[-10,0,0,-10]^T$	$[1,1,1,1]^T$	0	1557	5098	0.008005923
	$[0,0,0,0]^T$	$[1,1,1,1]^T$	0	1431	5099	0.007358045
Algorithm C	$[1,1,0,0]^T$	$[1,1,1,1]^T$	0	3041	35,243	0.015636489
	$[1,1,1,1]^T$	$[1,1,1,1]^T$	0	2867	34,570	0.014741800
	$[-10,10,-10,10]^T$	$[1,1,1,1]^T$	0	4608	39,849	0.023693831
	$[-10,-5,0,5]^T$	$[1,1,1,1]^T$	0	3842	37,147	0.019755143
	$[-10,0,0,-10]^T$	$[1,1,1,1]^T$	0	3174	35,253	0.016320360
	$[0,0,0,0]^T$	$[1,1,1,1]^T$	0	3051	35,254	0.015687908
Algorithm D	$[1,1,0,0]^T$	$[1,1,1,1]^T$	0	1615	15,973	0.008304153
	$[1,1,1,1]^T$	$[1,1,1,1]^T$	0	1435	15,312	0.007378613
	$[-10,10,-10,10]^T$	$[1,1,1,1]^T$	0	4145	21,660	0.021313136
	$[-10,-5,0,5]^T$	$[1,1,1,1]^T$	0	2569	17,483	0.013209517
	$[-10,0,0,-10]^T$	$[1,1,1,1]^T$	0	1748	15,992	0.008988025
	$[0,0,0,0]^T$	$[1,1,1,1]^T$	0	1625	15,993	0.008355572

Table 2
Numerical results of Problem 2.

Type	$\mathbf{x}_0$	$\mathbf{x}^*_{\text{final}}$	$f(\mathbf{x}^*_{\text{final}})$	E_f	E_G	R_E
Algorithm A	$[2,-2]^T$	$[0,-1]^T$	3	51,234	217,255	0.003200525
	$[0,-1]^T$	$[0,-1]^T$	3	47,189	217,255	0.002947838
	$[-2,-2]^T$	$[0,-1]^T$	3	53,675	217,255	0.003353011
	$[-0.5,-1]^T$	$[0,-1]^T$	3	47,189	217,255	0.002947838
	$[1,-1.5]^T$	$[0,-1]^T$	3	50,723	217,255	0.003168603
	$[1,-1]^T$	$[0,-1]^T$	3	47,189	217,255	0.002947838
Algorithm B	$[2,-2]^T$	$[0,-1]^T$	3	25,041	151,356	0.001564280
	$[0,-1]^T$	$[0,-1]^T$	3	18,995	151,356	0.001186594
	$[-2,-2]^T$	$[0,-1]^T$	3	24,472	151,356	0.001528736
	$[-0.5,-1]^T$	$[0,-1]^T$	3	20,475	151,356	0.001279048
	$[1,-1.5]^T$	$[0,-1]^T$	3	22,533	151,356	0.001407609
	$[1,-1]^T$	$[0,-1]^T$	3	21,978	151,356	0.001372938
Algorithm C	$[2,-2]^T$	$[0,-1]^T$	3	50,028	1,170,105	0.003125187
	$[0,-1]^T$	$[0,-1]^T$	3	45,983	1,170,105	0.002872501
	$[-2,-2]^T$	$[0,-1]^T$	3	52,469	1,170,105	0.003277673
	$[-0.5,-1]^T$	$[0,-1]^T$	3	45,983	1,170,105	0.002872501
	$[1,-1.5]^T$	$[0,-1]^T$	3	49,517	1,170,105	0.003093266
	$[1,-1]^T$	$[0,-1]^T$	3	45,983	1,170,105	0.002872501
Algorithm D	$[2,-2]^T$	$[0,-1]^T$	3	48,030	623,910	0.003000375
	$[0,-1]^T$	$[0,-1]^T$	3	43,985	623,910	0.002747688
	$[-2,-2]^T$	$[0,-1]^T$	3	50,475	623,910	0.003153111
	$[-0.5,-1]^T$	$[0,-1]^T$	3	43,985	623,910	0.002747688
	$[1,-1.5]^T$	$[0,-1]^T$	3	47,519	623,910	0.002968453
	$[1,-1]^T$	$[0,-1]^T$	3	43,985	623,910	0.002747688

Table 3
Numerical results of Problem 3.

Type	x_0	x^*_{final}	$f(x^*_{final})$	E_f	E_G	R_E
Algorithm A	$[10,-10]^T$	$[3,0.5]^T$	0	408,190	1,781,788	0.001020373
	$[9.997,-6.867]^T$	$[3,0.5]^T$	0	410,442	1,781,788	0.001026002
	$[0,-1]^T$	$[3,0.5]^T$	0	415,309	1,781,788	0.001038169
	$[1,1]^T$	$[3,0.5]^T$	0	216,860	1,140,046	0.000542096
	$[-2,2]^T$	$[3,0.5]^T$	0	219,484	1,140,046	0.000548655
	$[0,0]^T$	$[3,0.5]^T$	0	478,179	2,049,532	0.001195328
Algorithm B	$[10,-10]^T$	$[3,0.5]^T$	0	119,997	1,310,251	0.000299963
	$[9.997,-6.867]^T$	$[3,0.5]^T$	0	121,489	1,310,251	0.000303692
	$[0,-1]^T$	$[3,0.5]^T$	0	129,333	1,310,251	0.000323300
	$[1,1]^T$	$[3,0.5]^T$	0	107,219	723,603	0.000268021
	$[-2,2]^T$	$[3,0.5]^T$	0	105,842	723,603	0.000264579
	$[0,0]^T$	$[3,0.5]^T$	0	134,453	776,637	0.000336099
Algorithm C	$[10,-10]^T$	$[3.015,0.504]^{T a}$	0.0000376	100,002	128,430	0.000249980
	$[9.997,-6.867]^T$	$[3.015,0.504]^{T a}$	0.0000376	100,002	123,335	0.000249980
	$[0,-1]^T$	$[3.015,0.504]^{T a}$	0.0000376	100,001	111,165	0.000249978
	$[1,1]^T$	$[2.989,0.497]^{T a}$	0.0000211	100,001	199,532	0.000249978
	$[-2,2]^T$	$[2.989,0.497]^{T a}$	0.0000211	100,001	202,671	0.000249978
	$[0,0]^T$	$[2.989,0.497]^{T a}$	0.0000211	100,002	206,268	0.000249980
Algorithm D	$[10,-10]^T$	$[3.004,0.501]^{T a}$	0.00000255	386,183	2,610,857	0.000965361
	$[9.997,-6.867]^T$	$[3.004,0.501]^{T a}$	0.00000255	388,440	2,610,857	0.000971003
	$[0,-1]^T$	$[3.004,0.501]^{T a}$	0.00000255	393,307	2,610,857	0.000983169
	$[1,1]^T$	$[2.996,0.499]^{T a}$	0.00000257	257,134	2,110,006	0.000642771
	$[-2,2]^T$	$[2.996,0.499]^{T a}$	0.00000257	276,458	2,110,006	0.000691076
	$[0,0]^T$	$[3.004,0.501]^{T a}$	0.00000255	494,215	2,711,826	0.001235414

[a] *Remarks*: The final solution is a local solution.

Table 4
Numerical results of Problem 4.

Type	x_0	x^*_{final}	$f(x^*_{final})$	E_f	E_G	R_E
Algorithm A	$[10,10,10,10]^T$	$[0,0,0,0]^T$	0	1874	7248	1.17102×10^{-14}
	$[-10,-10,-10,-10]^T$	$[0,0,0,0]^T$	0	1928	7247	1.20476×10^{-14}
	$[10,-10,-10,10]^T$	$[0,0,0,0]^T$	0	1825	7248	1.14040×10^{-14}
	$[1,-1,-1,1]^T$	$[0,0,0,0]^T$	0	1742	7248	1.08853×10^{-14}
	$[-10,1,0,5]^T$	$[0,0,0,0]^T$	0	1807	7247	1.12915×10^{-14}
	$[0,0,0,0]^T$	$[0,0,0,0]^T$	0	1732	7243	1.08228×10^{-14}
Algorithm B	$[10,10,10,10]^T$	$[0,0,0,0]^T$	0	1160	5350	7.24855×10^{-15}
	$[-10,-10,-10,-10]^T$	$[0,0,0,0]^T$	0	1179	5349	7.36728×10^{-15}
	$[10,-10,-10,10]^T$	$[0,0,0,0]^T$	0	1131	5350	7.06734×10^{-15}
	$[1,-1,-1,1]^T$	$[0,0,0,0]^T$	0	1067	5350	6.66742×10^{-15}
	$[-10,1,0,5]^T$	$[0,0,0,0]^T$	0	1140	5349	7.12358×10^{-15}
	$[0,0,0,0]^T$	$[0,0,0,0]^T$	0	1061	5345	6.62992×10^{-15}
Algorithm C	$[10,10,10,10]^T$	$[0,0,0,0]^T$	0	2777	36,061	1.73528×10^{-14}
	$[-10,-10,-10,-10]^T$	$[0,0,0,0]^T$	0	2536	34,605	1.58468×10^{-14}
	$[10,-10,-10,10]^T$	$[0,0,0,0]^T$	0	2759	36,061	1.72403×10^{-14}
	$[1,-1,-1,1]^T$	$[0,0,0,0]^T$	0	2612	36,061	1.63217×10^{-14}
	$[-10,1,0,5]^T$	$[0,0,0,0]^T$	0	2420	34,605	1.51220×10^{-14}
	$[0,0,0,0]^T$	$[0,0,0,0]^T$	0	2342	34,594	1.46346×10^{-14}
Algorithm D	$[10,10,10,V10]^T$	$[0,0,0,0]^T$	0	2043	17,777	1.27662×10^{-14}
	$[-10,-10,-10,-10]^T$	$[0,0,0,0]^T$	0	1744	16,478	1.08978×10^{-14}
	$[10,-10,-10,10]^T$	$[0,0,0,0]^T$	0	2048	17,777	1.27974×10^{-14}
	$[1,-1,-1,1]^T$	$[0,0,0,0]^T$	0	1874	17,777	1.17102×10^{-14}
	$[-10,1,0,5]^T$	$[0,0,0,0]^T$	0	1620	16,478	1.01230×10^{-14}
	$[0,0,0,0]^T$	$[0,0,0,0]^T$	0	1542	16,458	9.63557×10^{-15}

4.1. Problem 1: Colville's function [19,27,28,31,42]

$$\begin{aligned}\min \quad & f(\mathbf{x}) = 100\left(x_2 - x_1^2\right)^2 + (1-x_1)^2 + 90\left(x_4 - x_3^2\right)^2 + (1-x_3)^2 \\ & \quad + 10.1\left[(x_2-1)^2 + (x_4-1)^2\right] + 19.8(x_2-1)(x_4-1) \\ \text{s.t.} \quad & -10 \leqslant x_i \leqslant 10, \quad x_i \text{ integer}, \quad i = 1,2,3,4.\end{aligned}$$

Table 5
Numerical results of Problem 5.

Type	$\mathbf{x}_0$	$\mathbf{x}^*_{\text{final}}$	$f(\mathbf{x}^*_{\text{final}})$	E_f	E_G	R_E
Algorithm A	$[0,\ldots,0]^T$	$[1,\ldots,1]^T$	0	211,831	682,050	1.95512×10^{-21}
	$[3,\ldots,3]^T$	$[1,\ldots,1]^T$	0	418,536	898,526	3.86292×10^{-21}
	$[-5,\ldots,-5]^T$	$[1,\ldots,1]^T$	0	217,435	682,050	2.00684×10^{-21}
	$[2,-2,\ldots,2,-2,2]^T$	$[1,\ldots,1]^T$	0	214,231	682,050	1.97727×10^{-21}
	$[3,-3,\ldots,3,-3,3]^T$	$[1,\ldots,1]^T$	0	510,907	1,006,018	4.71547×10^{-21}
	$[5,-5,\ldots,5,-5,5]^T$	$[1,\ldots,1]^T$	0	512,802	1,006,018	4.73296×10^{-21}
Algorithm B	$[0,\ldots,0]^T$	$[1,\ldots,1]^T$	0	171,072	444,101	1.57893×10^{-21}
	$[3,\ldots,3]^T$	$[1,\ldots,1]^T$	0	312,888	644,091	2.88783×10^{-21}
	$[-5,\ldots,-5]^T$	$[1,\ldots,1]^T$	0	176,624	444,101	1.63017×10^{-21}
	$[2,-2,\ldots,2,-2,2]^T$	$[1,\ldots,1]^T$	0	173,472	444,101	1.60108×10^{-21}
	$[3,-3,\ldots,3,-3,3]^T$	$[1,\ldots,1]^T$	0	191,402	563,646	1.76656×10^{-21}
	$[5,-5,\ldots,5,-5,5]^T$	$[1,\ldots,1]^T$	0	193,297	563,646	1.78405×10^{-21}
Algorithm C	$[0,\ldots,0]^T$	$[0,\ldots,0]^T$ [a]	24	532,603	3,031,547	4.91571×10^{-21}
	$[3,\ldots,3]^T$	$[1,\ldots,1]^T$	0	627,360	2,824,273	5.79277×10^{-21}
	$[-5,\ldots,-5]^T$	$[0,\ldots,0]^T$ [a]	24	538,156	3,031,547	4.96696×10^{-21}
	$[2,-2,\ldots,2,-2,2]^T$	$[0,\ldots,0]^T$ [a]	24	534,952	3,031,547	4.93739×10^{-21}
	$[3,-3,\ldots,3,-3,3]^T$	$[1,\ldots,1]^T$	0	678,295	2,920,682	6.26039×10^{-21}
	$[5,-5,\ldots,5,-5,5]^T$	$[1,\ldots,1]^T$	0	680,190	2,920,682	6.27788×10^{-21}
Algorithm D	$[0,\ldots,0]^T$	$[0,\ldots,0]^T$ [a]	24	182,636	1,493,376	1.68566×10^{-21}
	$[3,\ldots,3]^T$	$[1,\ldots,1]^T$	0	289,538	1,401,000	2.67232×10^{-21}
	$[-5,\ldots,-5]^T$	$[0,\ldots,0]^T$ [a]	24	188,189	1,493,376	1.73691×10^{-21}
	$[2,-2,\ldots,2,-2,2]^T$	$[0,\ldots,0]^T$ [a]	24	184,985	1,493,376	1.70734×10^{-21}
	$[3,-3,\ldots,3,-3,3]^T$	$[1,\ldots,1]^T$	0	339,380	1,493,460	3.13234×10^{-21}
	$[5,-5,\ldots,5,-5,5]^T$	$[1,\ldots,1]^T$	0	341,275	1,493,460	3.14983×10^{-21}

[a] *Remarks*: The final solution is a local solution.

This box constrained problem has 1.94481×10^5 feasible points. The global minimum solution is $\mathbf{x}^*_{\text{global}} = [1, 1, 1, 1]^T$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Six starting points were considered for the algorithms, namely $[1,1,0,0]^T$, $[1,1,1,1]^T$, $[-10,10,-10,10]^T$, $[-10,-5,0,5]^T$, $[-10,0,0,-10]^T$, and $[0,0,0,0]^T$. All discrete filled function algorithms succeeded in finding the global minimum from all starting points. A summary of the computational results is displayed in Table 1. Numerical results show that Algorithm B has the smallest total number of original function evaluations, and the average R_E is 0.008635805.

4.2. Problem 2: Goldstein and Price's function [15,27,28,42]

$$\begin{aligned} \min \quad & f(\mathbf{x}) = g(\mathbf{x})h(\mathbf{x}) \\ \text{s.t.} \quad & x_i = \frac{y_i}{1000} \quad -2000 \leqslant y_i \leqslant 2000, \quad y_i \text{ integer}, \quad i = 1, 2, \end{aligned}$$

where

$$g(\mathbf{x}) = 1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right),$$

and

$$h(\mathbf{x}) = 30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right).$$

This box constrained problem has 1.6008001×10^7 feasible points. The global minimum solution is $\mathbf{x}^*_{\text{global}} = [0, -1]^T$ with $f(\mathbf{x}^*_{\text{global}}) = 3$. Six starting points were considered in the computational tests, these being $[2,-2]^T$, $[0,-1]^T$, $[-2,-2]^T$, $[-0.5,-1]^T$, $[1,-1.5]^T$, and $[1,-1]^T$. A summary of the computational results is given in Table 2. All algorithms succeeded in finding the global minimum from all starting points, where Algorithm B is shown to be the most efficient method. This method succeeded in identifying the global minimum solution with an average of 22,249 function evaluations. The average R_E is 0.0013899.

4.3. Problem 3: Beale's function [25,27,28,31,42]

$$\begin{aligned} \min \quad & f(\mathbf{x}) = \left[1.5 - x_1(1 - x_2)\right]^2 + \left[2.25 - x_1\left(1 - x_2^2\right)\right]^2 + \left[2.625 - x_1\left(1 - x_2^3\right)\right]^2 \\ \text{s.t.} \quad & x_i = \frac{y_i}{1000} \quad -10000 \leqslant y_i \leqslant 10000, \quad y_i \text{ integer}, \quad i = 1, 2. \end{aligned}$$

This box constrained problem has 4.00040001×10^8 feasible points. The global minimum solution is $\mathbf{x}^*_{\text{global}} = [3, 0.5]^T$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Six starting points were considered in the tests: $[10,-10]^T$, $[9.997,-6.867]^T$, $[0,-1]^T$, $[1,1]^T$, $[-2,2]^T$, and

$[0,0]^T$. A summary of the computational results is shown in Table 3. Only Algorithms A and B succeeded in identifying the global minimum with the average number of function evaluations being 119722.2 and 358077.3, respectively. Note that Algorithm B is more efficient than Algorithm A, where the average R_E is 0.000299275, compared to 0.000895104. As for Algorithms C and D, both yielded local minimizers close to the global solution: $[3.015, 0.504]^T$, $[2.989, 0.497]^T$, $[3.004, 0.501]^T$, and $[2.996, 0.499]^T$. A possible reason for this failure to converge may be that our implementation calls on neighbourhood points in Step 3 in a different order to that in other implementations.

4.4. Problem 4: Powell's singular function [25,27,28,31,42]

$$\begin{aligned} \min \quad & f(\mathbf{x}) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4 \\ \text{s.t.} \quad & x_i = \frac{y_i}{1000} \quad - 10000 \leqslant y_i \leqslant 10000, \quad y_i \text{ integer}, \quad i = 1,2,3,4. \end{aligned}$$

This box constrained problem has 1.60032×10^{17} feasible points. The global minimum is at $\mathbf{x}^*_{\text{global}} = [0,0,0,0]^T$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Six starting points were used in the tests: $[10,10,10,10]^T$, $[-10,-10,-10,-10]^T$, $[10,-10,-10,10]^T$, $[1,-1,-1,1]^T$, $[-10,1,0,5]^T$, and $[0,0,0,0]^T$. All methods succeeded in identifying the global minimum. Table 4 summaries the computational results. Numerical experiments suggest that Algorithm B has the smallest total number of original function evaluations, and the average R_E is 7.01735×10^{-15}.

4.5. Problem 5: Rosenbrock's function [27,28,31,42]

$$\begin{aligned} \min \quad & f(\mathbf{x}) = \sum_{i=1}^{24} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \right] \\ \text{s.t.} \quad & -5 \leqslant x_i \leqslant 5, \quad x_i \text{ integer}, \quad i = 1,2,\ldots,25. \end{aligned}$$

This box constrained problem has 1.08347×10^{26} feasible points. The global minimum is at $\mathbf{x}^*_{\text{global}} = [1,\ldots,1]^T$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Six starting points were considered in the simulations: $[0,\ldots,0]^T$, $[3,\ldots,3]^T$, $[-5,\ldots,-5]^T$, $[2,-2,\ldots,2,-2,2]^T$, $[3,-3,\ldots,3,-3,3]^T$, and $[5,-5,\ldots,5,-5,5]^T$. All algorithms succeeded in identifying the global minimum for most of the starting points used. A summary of the computational results is displayed in Table 5. Clearly, Algorithm B has the least total number of original function evaluations and the average R_E is 1.87477×10^{-21}.

4.6. Comparison with literature results

Table 6 shows the average values of a number of original function evaluations for an algorithm to terminate and compares this with the results from the literature. Since these test problems were not solved in [43], we compare our numerical results with those in [27], [28], and [42] only. Recall that in our implementations of these algorithms, we construct a look-up table to

Table 6
A comparison of function evaluations.

Problem	Algorithm	Our implementations	Results in [28]	Results in [27]	Results in [42]
1	A	2440.17	4263.11		
	B	1679.5		3767.78	
	C	3430.5			85,705
	D	2189.5			
2	A	49533.17	111125.86		
	B	22,249		68196.29	
	C	48327.17			2,125,511
	D	46329.83			
3	A	366914.3	939209.57		
	B	119368.8		444887.71	
	C	100001.5[a]			4,861,560
	D	365956.2[a]			
4	A	1818	7337207.5		
	B	1123		6,731,232	
	C	2574.333			155,868,850
	D	1811.8333			
5	A	347623.7	320610.44		
	B	203125.8		305712.11	
	C	662038.3			6,282,030
	D	323397.7			

[a] *Remarks*: The final solution is a local solution.

store each objective function value computed so far to avoid repeated calculation of the objective function. Consequently, our implementations show a significantly lower number of function evaluations when compared to the results found in the literature. We note that in our implementation of the various algorithms, searches for a local minimum of the filled function may be initialized with different starting points than those used in the implementations published previously. This is because either the order in which the neighbourhood of $\mathbf{x}^*$ is to be tested is not specified or these starting points are not confined to the neighbourhood $N(\mathbf{x}^*)$ and are chosen randomly within the feasible region. This difference may influence the observed efficiency and accuracy of the algorithm. As can be seen from Table 6, Algorithm B is the most efficient method, yielding the lowest number of function evaluations for solving all test problems.

5. Concluding remarks

Various discrete filled function methods are reviewed in this paper. The fundamental idea behind the filled function concept is to introduce an auxiliary function to move from a current local minimizer to an improved point, if it exists. Interestingly, each filled function has its own termination and parameter updating criteria, though a generic algorithm is proposed here to try and capture their commonalities. Based on their theoretical properties, only discrete filled functions in [16] and [27] guarantee that a local minimizer of the filled function is also a local minimizer of the original function.

Discrete filled function methods have shown promising results in finding globally optimal solutions in several benchmark problems as demonstrated in the previous section, thus confirming the applicability, reliability, and efficiency of this relatively recent global optimization technique. Our intention is to adapt the technique to complex mixed discrete optimization problems where individual objective function evaluations are computationally expensive. Methods requiring the least number of function evaluations are important in this context [35,36].

Acknowledgements

The authors thank Ryan Loxton from Curtin University of Technology for his constructive feedback on some algorithms in this manuscript. Special thanks also goes to David Packer from Curtin University of Technology for his valuable comments on the original manuscript.

References

[1] J. Bang-Jensen, G. Gutin, A. Yeo, When the greedy algorithm fails, Discrete Optimization 1 (2004) 121–127.
[2] J. Cai, G. Thierauf, Evolution strategies for solving discrete optimization problems, Advances in Engineering Software 25 (1996) 177–183.
[3] D. Chakrabarty, N.R. Devanur, V.V. Vazirani, New geometry-inspired relaxations and algorithms for the Metric Steiner Tree Problem, in: A. Lodi, A. Panconesi, G. Rinaldi (Eds.), Integer Programming and Combinatorial Optimization: 13th International Conference, IPCO 2008 Bertinoro, 2008 Proceedings, Italy, May 26–28, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 344–358.
[4] V. Chvatal, A greedy heuristic for the set-covering problem, Mathematics of Operations Research 4 (3) (1979) 233–235.
[5] L.C.W. Dixon, G.P. Szego (Eds.), Towards Global Optimization, Elsevier Science Limited: North-Holland, 1975.
[6] G. Dobson, Worst-case analysis of greedy heuristics for integer programming with nonnegative data, Mathematics of Operations Research 7 (4) (1982) 515–531.
[7] A. Etzel, Mixed discrete optimization of multiple-valued systems, in: 27th International Symposium on Multiple-Valued Logic Proceedings, Antigonish, Canada, May 28–30, 1997, pp. 259–264.
[8] A. Federgruen, H. Groenevelt, The greedy procedure for resource allocation problems: necessary and sufficient conditions for optimality, Operations Research 34 (6) (1986) 909–918.
[9] M.L. Fisher, The Lagrangian relaxation method for solving integer programming problems, Management Science 50 (12) (2004) 1861–1871.
[10] R. Fukasawa, M. Goycoolea, On the exact separation of mixed integer knapsack cuts, in: M. Fischetti, D.P. Williamson (Eds.), Integer Programming and Combinatorial Optimization: 12th International Conference, IPCO 2007 Ithaca, 2007 Proceedings, New York, June 25–27, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 225–239.
[11] R. Ge, A filled function method for finding a global minimizer of a function of several variables, Mathematical Programming 46 (1) (1990) 191–204.
[12] R. Ge, C. Huang, A continuous approach to nonlinear integer programming, Applied Mathematics and Computation 34 (1) (1989) 39–60.
[13] A.M. Geoffrion, Lagrangian relaxation for integer programming, Mathematical Programming Study 2 (1974) 82–114.
[14] F. Glover, Tabu search and adaptive memory programming – advances applications, and challenges, in: R.S. Barr, R.V. Helgason, J.L. Kennington (Eds.), Interfaces in Computer Science and Operations Research: Advances in Metaheuristics, Optimization, and Stochastic Modeling Technologies, Kluwer Academic Publishers, 1996, pp. 1–75.
[15] A.A. Goldstein, J.F. Price, On descent from local minima, Mathematics of Computation 25 (115) (1971) 569–574.
[16] Y.H. Gu, Z.Y. Wu, A new filled function method for nonlinear integer programming problem, Applied Mathematics and Computation 173 (2) (2006) 938–950.
[17] O. Günlük, J. Linderoth, Perspective relaxation of mixed integer nonlinear programs with indicator variables, in: A. Lodi, A. Panconesi, G. Rinaldi (Eds.), Integer Programming and Combinatorial Optimization: 13th International Conference, IPCO 2008 Bertinoro, 2008 Proceedings, Italy, May 26–28, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 1–16.
[18] O.K. Gupta, A. Ravindran, Branch and bound experiments in convex nonlinear integer programming, Management Science 31 (12) (1985) 1533–1546.
[19] W. Hock, K. Schittkowski, Test Examples for Nonlinear Programming Codes, Springer-Verlag, New York, 1980.
[20] W.J. Lee, A.V. Cabot, M.A. Venkataramanan, A branch and bound algorithm for solving separable convex integer programming problems, Computers and Operations Research 21 (9) (1994) 1011–1024.
[21] K. Madsen, J. Žilinskas, Testing Branch-and-Bound Methods for Global Optimization, IMM Department of Mathematical Modelling, Technical University of Denmark, 2000.
[22] R.E. Marsten, T.L. Morin, A hybrid approach to discrete mathematical programming, Mathematical Programming 14 (1978) 21–40.
[23] L. Michel, P. Van Hentenryck, A simple tabu search for warehouse location, European Journal of Operational Research 157 (2004) 576–591.
[24] C. Mohan, H.T. Nguyen, A controlled random search technique incorporating the simulated annealing concept for solving integer and mixed integer global optimization problems, Computational Optimization and Applications 14 (1999) 103–132.
[25] J.J. Moré, B.S. Garbow, K.E. Hillstrom, Testing unconstrained optimization software, ACM Transactions on Mathematical Software 7 (1) (1981) 17–41.

[26] C.K. Ng, D. Li, L.S. Zhang, Filled function approaches to nonlinear integer programming: a survey, in: S.H. Hou, X.M. Yang, G.Y. Chen (Eds.), Frontiers in Optimization and Control, vol. 20, Kluwer Academic Publishers, 2005, pp. 1–20.
[27] C.K. Ng, D. Li, L.S. Zhang, Discrete global descent method for discrete global optimization and nonlinear integer programming, Journal of Global Optimization 37 (3) (2007) 357–379.
[28] C.K. Ng, L.S. Zhang, D. Li, W.W. Tian, Discrete filled function method for discrete global optimization, Computational Optimization & Applications 31 (1) (2005) 87–115.
[29] M. Rim, R. Jain, Lower-bound performance estimation for the high-level synthesis scheduling problem, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 13 (4) (1994) 51–458.
[30] S.L. Rosen, C.M. Harmonosky, An improved simulated annealing simulation optimization method for discrete parameter stochastic systems, Computers and Operations Research 32 (2005) 343–358.
[31] K. Schittkowski, More Test Examples for Nonlinear Programming Codes, Springer-Verlag, New York, 1987.
[32] Y. Shang, L. Zhang, A filled function method for finding a global minimizer on global integer optimization, Journal of Computational and Applied Mathematics 181 (1) (2005) 200–210.
[33] Y. Shang, L. Zhang, Finding discrete global minima with a filled function for integer programming, European Journal of Operational Research 189 (1) (2008) 31–40.
[34] N. Turkkan, Discrete optimization of structures using a floating-point genetic algorithm, in: Annual Conference of the Canadian Society for Civil Engineering, Moncton, Canada, June 4–7, 2003.
[35] S.F. Woon, Global Algorithms for Nonlinear Discrete Optimization and Discrete-Valued Optimal Control Problems, Ph.D. Thesis, Department of Mathematics and Statistics, Curtin University of Technology, 2009.
[36] S.F. Woon, V. Rehbock, R.C. Loxton, Global optimization method for continuous-time sensor scheduling, Special Issue of Dynamics and Systems Theory 10 (2) (2010) 175–188.
[37] S.J. Wu, P.T. Chow, Steady-state genetic algorithms for discrete optimization of trusses, Computers and Structures 56 (6) (1995) 979–991.
[38] Z. Wu, B.W. Wah, The theory of discrete Lagrange multipliers for nonlinear discrete optimization, in: J. Jaffar (Ed.), Principles and Practice of Constraint Programming: 5th International Conference, CP 1999, 1999 Proceedings, Alexandria, Virginia, October 11–14, Springer-Verlag, Berlin, Heidelberg, 1999, pp. 28–42.
[39] Z. Wu, B.W. Wah, An efficient global-search strategy in discrete Lagrangian methods for solving hard satisfiability problems, in: Proceedings of the 17th National Conference on Artificial Intelligence, Austin, Texas, July 30–August 3, 2000, pp. 310–315.
[40] X. Xu, P.J. Antsaklis, Optimal control of switched autonomous systems, in: Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, December 10–13, vol. 4, 2002, pp. 4401–4406.
[41] X.Q. Yang, C.J. Goh, A nonlinear Lagrangian function for discrete optimization problems, in: A. Migdalas, A. Pardalos, P. Varbrand, K. Holmqvist (Eds.), From Local to Global Optimization, Kluwer Academic Publishers, The Netherlands, 2000, pp. 291–304.
[42] Y. Yang, Y. Liang, A new discrete filled function algorithm for discrete global optimization, Journal of Computational and Applied Mathematics 202 (2) (2007) 280–291.
[43] Y. Yang, Z. Wu, F. Bai, A filled function method for constrained nonlinear integer programming, Journal of Industrial and Management Optimization 4 (2) (2008) 353–362.
[44] Y. Yang, L. Zhang, A gradually descent method for discrete global optimization, Journal of Shanghai University (English Edition) 11 (1) (2007) 39–44.
[45] A. Zanette, M. Fischetti, E. Balas, Can pure cutting plane algorithms work? in: A. Lodi, A. Panconesi, G. Rinaldi (Eds.), Integer Programming and Combinatorial Optimization: 13th International Conference, IPCO 2008 Bertinoro, Italy, May 26–28, 2008 Proceedings, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 416–434.
[46] W. Zhu, An approximate algorithm for nonlinear integer programming, Applied Mathematics and Computations 93 (2-3) (1998) 183–193.