On the Modification of the Discrete Filled Function Algorithm for Nonlinear Discrete Optimization

S.F. Woon, S.Karim, *Member, IAENG,* M.S.A Mohamad, L. Ryan, and V. Rehbock

***Abstract*—The discrete filled function method (DFFM) is a global optimization method for searching for the best solution amongst multiple local optima. This method consists of two phases: in the first phase, an ordinary descent method is used to find a local minimum; in the second phase, an auxiliary function, called a filled function, is introduced that has a maximizer at the current local minimum, so that minimizing the filled function leads to improved points. Once an improved point is found, it can serve as a starting point for the next local search. In this paper, we consider a standard discrete filled function algorithm in the literature and propose a modification to increase its efficiency. Three numerical examples are given to demonstrate the proposed modification's potential in solving large scale discrete optimization problems.**

Index Terms- Global optimization, discrete filled function, optimal control, mixed discrete optimization, heuristic.

I. Introduction

The DFFM is one of the more recently developed global optimization heuristic approach in solving discrete optimization problems [1]. The main idea is as follows: An ordinary descent method is first applied to identify a local minimum. Once found, the DFFM is used to suggest sequential improvement of local optima strategy through an auxiliary function to escape from one local minima to a better one. Consequently, the local minimum of the original function becomes a local maximum of the auxiliary function. The search for the improved minimum in a lower basin continues until the parameter of the discrete filled function is satisfied. The final point found is expected to be the global minimum.

The first filled function was introduced by Ge [2] to solve continuous global optimization problems in the late 1980s. Then, Zhu [3] initiated a discrete analogue of the continuous filled function method which overcame the difficulties encountered in using a continuous approximation of the discrete optimization problem. However, the filled function proposed by Zhu contains an exponential term, which causes numerical difficulties such as numerical overflow [4]. Since the introduction of the original discrete filled function by Zhu, several new types of DFFM with improved theoretical properties have been proposed, such as those in [4]–[8], to enhance computational efficiency. In spite of computational efficiency, exist researcher applied DFFM in some area such in designing sparse filter [9]. A comprehensive survey of several DFFM in the literature has been given in [10]. The survey showed that the DFFM developed in [4] seems to be the most reliable one since it guarantees that a local minimizer of the filled function is also a local minimizer of the original function. Other filled functions do not share this property where they can only guarantee that a local minimizer of the filled function is an improved point over the current local minimizer of the original function. The goal of this paper is to propose an improved filled function algorithm based on the work in [4].

The remainder of the paper is organized as follows. An overview of discrete optimization is presented in the next section, followed by a brief review of the discrete filled function algorithm in [4]. Then, the modified algorithm is proposed in Section IV. The modification on some benchmark test problems is tested in Section V. Comparison and further analysis of both algorithms are discussed before concluding the paper.

Manuscript received December 21, 2020; revised June 01, 2021. This work was supported by Universiti Utara Malaysia under LEADS Research Grant (S/O Code: 12408).

Woon Siew Fang is a pensioner and currently stay in Penang, Malaysia.(e-mail: woonsiewfang@yahoo.com)

Sharmila Karim is a senior lecturer of School of Quantitative Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia. (e-mail: mila@uum.edu.my)

Mohd Saiful Adli Mohamad is a senior lecturer of School of Quantitative Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia. (e-mail:msadli@uum.edu.my)

Ryan Loxton is a Professor of Department of Mathematics and Statistics, Curtin University,GPO Box U1987, Perth Western, 6845 Australia.(e-mail: R.Loxton@curtin.edu.au)

Rehbock Volker is a Professor of Department of Mathematics and Statistics, Curtin University,GPO Box U1987, Perth Western, 6845 Australia.(e-mail:rehbock@maths.curtin.edu.au)

II. Preliminary Concepts

Consider the following nonlinear discrete optimization problem:

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

where $X = \{\mathbf{x} \in \mathbb{Z}^n : \mathbf{x}_{i,\min} \leq \mathbf{x}_i \leq \mathbf{x}_{i,\max},\ i = 1, \ldots, n\}$, $\mathbb{Z}^n$ 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. Since X is bounded, there exists a constant $\mathcal{K}$ such that

$$1 \leq \max_{\substack{\mathbf{x}_1, \mathbf{x}_2 \in X \\ \mathbf{x}_1 \neq \mathbf{x}_2}} \| \mathbf{x}_1 - \mathbf{x}_2 \| \leq \mathcal{K} < \infty, \tag{2}$$

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

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

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

Definition 2: For any $\mathbf{x} \in X$, the *neighbourhood* of $\mathbf{x}$ is

defined by

$$N(\mathbf{x}) = \{\mathbf{w} \in X : \ \mathbf{w} = \mathbf{x} \pm \mathbf{e}_i, \ \ i = 1, \ldots, n\},$$

where $\mathbf{e}_i$ denotes the i-th standard unit basis vector of $\mathbb{R}^n$ with the i-th component equal to one and all other components equal to zero.

Definition 3: The set of 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 *steepest descent direction* of f at $\mathbf{x}$ if it is a descent direction and $f(\mathbf{x}+\mathbf{d}^*) \leq f(\mathbf{x}+\mathbf{d})$ for all $\mathbf{d} \in \mathcal{D}(\mathbf{x})$.

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

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

Definition 8: $\mathbf{x}$ is a *vertex* of X if $\mathbf{x} - \mathbf{d} \notin X$ for each $\mathbf{d} \in D(\mathbf{x})$. Let $\tilde{X}$ denote the set of vertices of X.

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

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

Definition 11: For a given local minimizer $\mathbf{x}^*$, define the discrete sets

$$S_L(\mathbf{x}^*) = \{\mathbf{x} \in X : f(\mathbf{x}) < f(\mathbf{x}^*)\}$$

and

$$S_U(\mathbf{x}^*) = \{\mathbf{x} \in X : f(\mathbf{x}) \geq f(\mathbf{x}^*)\}.$$

Note that $S_L(\mathbf{x}^*)$ contains the points lower than $\mathbf{x}^*$, while $S_U(\mathbf{x}^*)$ contains the points higher than $\mathbf{x}^*$.

Let $\mathbf{x}^*$ be a local minimizer of f. In [4], the discrete filled function $G_{\mu,\rho,\mathbf{x}^*}$ at $\mathbf{x}^*$ is defined as follows:

$$G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}) = A_\mu(f(\mathbf{x}) - f(\mathbf{x}^*)) - \rho \parallel \mathbf{x} - \mathbf{x}^* \parallel, \qquad (3)$$

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

where $c \in (0,1)$ is a constant, $\omega > 0$ is a sufficiently small constant, $\rho > 0$, and $\mu \in (0,1)$ are adjustable parameters. It can be shown that the function $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x})$ is a discrete filled function when certain conditions on the parameters μ and ρ are satisfied, as revealed by the following properties proved in [4]:

- $\mathbf{x}^*$ is a strict local maximizer of $G_{\mu,\rho,\mathbf{x}^*}$ if $\rho > 0$ and $0 < \mu < \min\{1, \rho/\mathcal{L}\}$.
- If $\mathbf{x}^*$ is a global minimizer of f, then $G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}) < 0$ for all $\mathbf{x} \in X \setminus \mathbf{x}^*$.
- Let $\bar{\mathbf{d}} \in \mathcal{D}(\bar{\mathbf{x}})$ be a feasible direction at $\bar{\mathbf{x}} \in S_U(\mathbf{x}^*)$ such that

$$\parallel \bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^* \parallel > \parallel \bar{\mathbf{x}} - \mathbf{x}^* \parallel .$$

If $\rho > 0$ and $0 < \mu < \min\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\}$, 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 local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ and suppose that there exists a feasible direction $\bar{\mathbf{d}} \in \mathcal{D}(\acute{\mathbf{x}})$ such that $\parallel \acute{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^* \parallel > \parallel \acute{\mathbf{x}} - \mathbf{x}^* \parallel$. If $\rho > 0$ is sufficiently small and $0 < \mu < \min\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\}$, 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 $0 < \mu < \min\{1, \frac{\rho}{2\mathcal{K}^2\mathcal{L}}\}$. Then, $\mathbf{x}^{**} \in X \setminus \tilde{X}$ is a local minimizer of f with $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$ if and only if $\mathbf{x}^{**}$ is a local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$.

III. The Standard Algorithm

The standard algorithm described in [4] consists of two phases: the first phase involves finding a local minimum of f; the second phase involves finding an improved point from which the local search can be restarted. These phases are stated in details as Algorithms 1 and 2 below.

Algorithm 1: **(Local search)**

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

Algorithm 2: **(Search for an improved point)**

1) Initialize $\mathbf{x}_0 \in X$, ρ_0, μ_0, $\rho_L > 0$, $0 < \hat{\rho} < 1$, and $0 < \hat{\mu} < 1$.
   Set $\rho := \rho_0$ and $\mu := \mu_0$.
2) Starting from $\mathbf{x}_0$, minimize $f(\mathbf{x})$ using Algorithm 1 to obtain a local minimizer $\mathbf{x}^*$ of f.
3) (a) List the neighbouring points of $\mathbf{x}^*$ as $N(\mathbf{x}^*) = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q\}$. Set $\ell := 1$.
   (b) Set the current point, $\mathbf{x}_c := \mathbf{w}_\ell$.
4) (a) If there exists a direction $\mathbf{d} \in \mathcal{D}(\mathbf{x}_c)$ such that $f(\mathbf{x}_c+\mathbf{d}) < f(\mathbf{x}^*)$, set $\mathbf{x}_0 := \mathbf{x}_c+\mathbf{d}$ and go to Step 2. Otherwise, go to (b) below.
   (b) Let

$$\mathcal{D}_1 = \{\mathbf{d} \in \mathcal{D}(\mathbf{x}_c) : \ f(\mathbf{x}_c + \mathbf{d}) < f(\mathbf{x}_c)$$

and

$$G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d}) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c)\}.$$

   If $\mathcal{D}_1 \neq \emptyset$, set $\mathbf{d}^* := \arg\min_{\mathbf{d} \in \mathcal{D}_1}\{f(\mathbf{x}_c + \mathbf{d}) + G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d})\}$, set $\mathbf{x}_c := \mathbf{x}_c + \mathbf{d}^*$ and go to (a) above. Otherwise, go to (c) below.
   (c) Let $\mathcal{D}_2 = \{\mathbf{d} \in \mathcal{D}(\mathbf{x}_c) : \ G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d}) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c)\}$. If $\mathcal{D}_2 \neq \emptyset$, set $\mathbf{d}^* := \arg\min_{\mathbf{d} \in \mathcal{D}_2}\{G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d})\}$, set $\mathbf{x}_c := \mathbf{x}_c + \mathbf{d}^*$ and go to (a) above.
   Otherwise, go to Step 5.
5) Let $\acute{\mathbf{x}} = \mathbf{x}_c$ be the local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ obtained from Step 4.

(a) If $\acute{\mathbf{x}} \in \tilde{X}$, set $\ell := \ell + 1$. If $\ell > q$, go to Step 6. Otherwise, go to Step 3(b).
(b) If $\acute{\mathbf{x}} \notin \tilde{X}$, reduce μ by setting $\mu := \hat{\mu}\mu$ and go to Step 4(b).

6) Reduce ρ by setting $\rho := \hat{\rho}\rho$. If $\rho < \rho_L$, terminate the algorithm. The current $\mathbf{x}^*$ is taken as a global minimizer of the problem. Otherwise, set $\ell := 1$ and go to Step 3(b).

The discrete filled function approach in Algorithm 2 above can be explained as follows. First, an initial point is chosen before applying a local search to find a discrete local minimizer. Then, the neighborhood of the discrete local minimizer is set up. Next, a discrete filled function is constructed and the local minimizer of the original function becomes the local maximizer of the discrete filled function. By minimizing the discrete filled function, a discrete local minimizer of the discrete filled function is found. The discrete local minimizer of the discrete filled function is examined whether it is an improved point or a corner point. If it is an improved point, the local minimizer of the discrete filled function transforms to be a new starting point to minimize the original function in such a way that an improved local minimizer can be found. In most situations, the local minimizer of discrete filled function is also a local minimizer of the original function when some standard assumption are satisfied. On the other hand, if the discrete local minimizer of the discrete filled function is a corner point, the next point in the neighbourhood in Step 3 is chosen to minimize the discrete filled function, until all points in the neighbourhood are tested. Then, the parameter ρ is reduced and the search of the discrete local minimizer of the discrete filled function is repeated. However, if the local minimizer of the discrete filled function is neither an improved point or corner point, the parameter μ is reduced and a new discrete filled function is constructed at the current point under the new parameter setting. The parameters reduction continues until they reached their presetting values, and the best solution found is treated as the global minimizer.

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 yielding 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 both f and $G_{\mu,\rho,\mathbf{x}^*}$ simultaneously exists. If such a direction does not exist, the algorithm uses the steepest descent direction for $G_{\mu,\rho,\mathbf{x}^*}$.

IV. The Modified Algorithm

We replace the neighbourhood $N(\mathbf{x}^*)$ in Step 3 of Algorithm 2 with a set of randomly chosen points from X. Then, an additional step is added to test whether any one of these random points happens to be an improved point. The motivation for this modification is to search for improved points more efficiently by choosing points that give a broader coverage of X, similar to the approaches proposed in [5], [11], [12]. In the standard approach, the initial points are chosen as the neighbouring points of the current local solution. The modified algorithm is described formally below.

Algorithm 3: **(Modified filled function algorithm with random starting points)**

1) Initialize $\mathbf{x}_0 \in X$, ρ_0, μ_0, $\rho_L > 0$, $0 < \hat{\rho} < 1$, and $0 < \hat{\mu} < 1$.
Let $\rho := \rho_0$ and $\mu := \mu_0$.
2) Starting from $\mathbf{x}_0$, minimize $f(\mathbf{x})$ using Algorithm 1 to obtain a local minimizer $\mathbf{x}^*$ of f.
3) Let $M = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_q\}$, where $\mathbf{w}_\ell$, $\ell = 1, \ldots, q$, are randomly chosen from X and $q = 2n$.
4) (a) Set $\ell := 1$.
(b) If $f(\mathbf{w}_\ell) < f(\mathbf{x}^*)$, set $\mathbf{x}_0 := \mathbf{w}_\ell$ and go to Step 2. Otherwise, go to (c) below.
(c) Set $\ell := \ell + 1$. If $\ell \leq q$, go to (b) above. Otherwise, set $\ell := 1$ and go to (d) below.
(d) Set the current point $\mathbf{x}_c := \mathbf{w}_\ell$.
5) (a) If there exists a direction $\mathbf{d} \in \mathcal{D}(\mathbf{x}_c)$ such that $f(\mathbf{x}_c + \mathbf{d}) < f(\mathbf{x}^*)$, set $\mathbf{x}_0 := \mathbf{x}_c + \mathbf{d}$ and go to Step 2. Otherwise, go to (b) below.
(b) Let

$$\mathcal{D}_1 = \{\mathbf{d} \in \mathcal{D}(\mathbf{x}_c) : \ f(\mathbf{x}_c + \mathbf{d}) < f(\mathbf{x}_c) \text{ and } G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d}) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c)\}.$$

If $\mathcal{D}_1 \neq \emptyset$, set $\mathbf{d}^* := \arg\min_{\mathbf{d} \in \mathcal{D}_1}\{f(\mathbf{x}_c + \mathbf{d}) + G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d})\}$, set $\mathbf{x}_c := \mathbf{x}_c + \mathbf{d}^*$ and go to (a) above.
Otherwise, go to (c) below.
(c) Let $\mathcal{D}_2 = \{\mathbf{d} \in \mathcal{D}(\mathbf{x}_c) : \ G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d}) < G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c)\}$.
If $\mathcal{D}_2 \neq \emptyset$, set $\mathbf{d}^* := \arg\min_{\mathbf{d} \in \mathcal{D}_2}\{G_{\mu,\rho,\mathbf{x}^*}(\mathbf{x}_c + \mathbf{d})\}$, set $\mathbf{x}_c := \mathbf{x}_c + \mathbf{d}^*$ and go to (a) above.
Otherwise, go to Step 6.
6) Let $\acute{\mathbf{x}} = \mathbf{x}_c$ be the local minimizer of $G_{\mu,\rho,\mathbf{x}^*}$ obtained from Step 5.
(a) If $\acute{\mathbf{x}} \in \tilde{X}$, set $\ell := \ell + 1$. If $\ell > q$, go to Step 7. Otherwise, go to Step 4(d).
(b) If $\acute{\mathbf{x}} \notin \tilde{X}$, reduce μ by setting $\mu := \hat{\mu}\mu$ and go to Step 5(b).
7) Reduce ρ by setting $\rho := \hat{\rho}\rho$. If $\rho < \rho_L$, terminate the algorithm. The current $\mathbf{x}^*$ is taken as a global minimizer of the problem. Otherwise, set $\ell := 1$ and go to Step 4(d).

V. Numerical Results

We tested both algorithms on Rosenbrock's and Colville's functions on FORTRAN. The adjustable parameters μ and ρ are both initialized to 0.1. The parameter μ is reduced if $\acute{\mathbf{x}}$ is neither a vertex nor an improved point by setting $\mu := \mu/10$. The other parameter, ρ, is reduced when all searches in minimizing the filled function end up at vertices. Note that we set $\rho_L = 0.001$ in the numerical computation.

Problem 1: Colville's Function [13]

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

TABLE I
RESULTS OF ALGORITHM 2 - COLVILLE'S FUNCTION.

$\mathbf{x}_0$	E_f	E_G	R_E
$[1,1,0,0]^\top$	1426	5097	0.007332336
$[1,1,1,1]^\top$	1422	5076	0.007311768
$[-10,10,-10,10]^\top$	2674	5979	0.013749415
$[-10,-5,0,5]^\top$	1567	5134	0.008057342
$[-10,0,0,-10]^\top$	1557	5098	0.008005923
$[0,0,0,0]^\top$	1431	5099	0.007358045

TABLE II
RESULTS OF ALGORITHM 3 - COLVILLE'S FUNCTION.

$\mathbf{x}_0$	E_f	E_G	R_E	N_T
$[1,1,0,0]^\top$	1092	2812	0.005614944	13
$[1,1,1,1]^\top$	1030	2387	0.005296147	1
$[-10,10,-10,10]^\top$	1106	2659	0.005686931	7
$[-10,-5,0,5]^\top$	1542	3759	0.007928795	15
$[-10,0,0,-10]^\top$	1135	3101	0.005836046	2
$[0,0,0,0]^\top$	954	3010	0.004905364	12

The Colville's function has 1.94481×10^5 feasible points and a global minimum $\mathbf{x}^*_{\text{global}} = [1,1,1,1]^\top$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Six starting points are considered in Algorithm 2, namely $[1,1,0,0]^\top$, $[1,1,1,1]^\top$, $[-10,10,-10,10]^\top$, $[-10,-5,0,5]^\top$, $[-10,0,0,-10]^\top$, and $[0,0,0,0]^\top$. The algorithm succeeded in finding the global minimum from all starting points, as displayed in Table I. The total number of original function evaluations, the total number of discrete filled function evaluations, and the ratio of the number of original function evaluations to the total number of feasible points are denoted in the table by E_f, E_G, and R_E, respectively.

On the other hand, Algorithm 3 succeeds in determining the global solution of Colville's function much more efficiently with an average $E_f = 1143.2$, compared with $E_f = 1679.5$ obtained by Algorithm 2, which is a reduction of 31.9% in average total number of original function evaluations (see Table IX). However, several starting points used in Algorithm 3 were unable to determine the global solution for the initial choice of the random set M. In these cases, we repeated the application of the algorithm several times until the global optimum was obtained (note that the random set M changes with each new application). Note that the E_f values in Table II show the number of function evaluations recorded for the successful application of the algorithm only. The number of required attempts before reaching the global solution is denoted by N_T in Table II.

Problem 2: Goldstein and Price's Function [13]

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

where

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

and

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

The Goldstein and Price's function has 1.6008001×10^7 feasible points, with the global minimum solution $f(\mathbf{x}^*_{\text{global}}) = 3$ when $\mathbf{x}^*_{\text{global}} = [0,-1]^\top$. Six initial points were used in the simulation tests: $[2,-2]^\top$, $[0,-1]^\top$, $[-2,-2]^\top$, $[-0.5,-1]^\top$, $[1,-1.5]^\top$, and $[1,-1]^\top$. The computational results are summarized in Table III. Algorithm 2 converges to the global minimum solution with an average of 22249 function evaluations. The average R_E is 0.0013899.

TABLE III
RESULTS OF ALGORITHM 2 - GOLDSTEIN AND PRICE'S FUNCTION.

$\mathbf{x}_0$	E_f	E_G	R_E
$[2,-2]^\top$	25041	151356	0.001564280
$[0,-1]^\top$	18995	151356	0.001186594
$[-2,-2]^\top$	24472	151356	0.001528736
$[-0.5,-1]^\top$	20475	151356	0.001279048
$[1,-1.5]^\top$	22533	151356	0.001407609
$[1,-1]^\top$	21978	151356	0.001372938

TABLE IV
RESULTS OF ALGORITHM 3 - GOLDSTEIN AND PRICE'S FUNCTION.

$\mathbf{x}_0$	E_f	E_G	R_E	N_T
$[2,-2]^\top$	20820	66975	0.001300600	1
$[0,-1]^\top$	18602	87844	0.001162044	1
$[-2,-2]^\top$	21440	64318	0.001339330	1
$[-0.5,-1]^\top$	17625	53214	0.001101012	1
$[1,-1.5]^\top$	20309	66975	0.001268678	1
$[1,-1]^\top$	11419	33463	0.000713331	1

TABLE V
RESULTS OF ALGORITHM 2 - ROSENBROCK'S FUNCTION FOR $n = 5$.

$\mathbf{x}_0$	E_f	E_G	R_E
$[-5,-5,-5,-5,-5]^\top$	1436	4475	0.45952
$[-4,-4,-4,-4,-4]^\top$	1435	4475	0.45920
$[-3,-3,-3,-3,-3]^\top$	1395	4475	0.44640
$[-2,-2,-2,-2,-2]^\top$	1354	4475	0.43328
$[-1,-1,-1,-1,-1]^\top$	1314	4475	0.42048
$[0,0,0,0,0]^\top$	1274	4475	0.40768
$[1,1,1,1,1]^\top$	1252	4415	0.40064
$[2,2,2,2,2]^\top$	1271	4415	0.40672
$[3,3,3,3,3]^\top$	1646	5720	0.52672
$[4,4,4,4,4]^\top$	1666	5720	0.53312
$[5,5,5,5,5]^\top$	1664	5720	0.53248

Unlike the first problem, Algorithm 3 not only show its efficiency in minimizing the Goldstein and Price's function but has no reliability issue where this algorithm succeeded in finding the global solution with only one attempt for each of the initial points tested. The average function evaluations and average ratio total number of function evaluations to total number of feasible points are 18369.16667 and 0.001147499, respectively. It is an improvement of 17.44% compared to Algorithm 2. Numerical results of Algorithm 3 in solving Goldstein and Price's function is represented by Table IV. A comparison between the two algorithms can be found in Table IX.

Problem 3: Rosenbrock's Function [13]

$$\min \ f(\mathbf{x}) = \sum_{i=1}^{n} \Big[100\big(x_{i+1} - x_i^2\big)^2 + \big(1 - x_i\big)^2\Big],$$
$$\text{s.t.} \quad -5 \le x_i \le 5, \quad x_i \text{ integer}, \quad i = 1,2,\ldots,n.$$

Table V gives the numerical results from implementing Algorithm 2 to minimize Rosenbrock's function with $n = 5$ and 1.61015×10^5 feasible points. The global

TABLE VI
RESULTS OF ALGORITHM 3 - ROSENBROCK'S FUNCTION FOR $n = 5$.

$\mathbf{x}_0$	E_f	E_G	R_E	N_T
$[-5,-5,-5,-5,-5]^\top$	937	2115	0.29984	1
$[-4,-4,-4,-4,-4]^\top$	1219	2662	0.39008	1
$[-3,-3,-3,-3,-3]^\top$	1445	3279	0.46240	1
$[-2,-2,-2,-2,-2]^\top$	923	2060	0.29536	1
$[-1,-1,-1,-1,-1]^\top$	1032	2409	0.33024	2
$[0,0,0,0,0]^\top$	936	2447	0.29952	2
$[1,1,1,1,1]^\top$	820	1961	0.26240	1
$[2,2,2,2,2]^\top$	871	2165	0.27872	1
$[3,3,3,3,3]^\top$	1233	2904	0.39456	3
$[4,4,4,4,4]^\top$	1356	3112	0.43392	3
$[5,5,5,5,5]^\top$	980	2269	0.31360	1

TABLE VII
RESULTS OF ALGORITHM 2 - ROSENBROCK'S FUNCTION FOR $n = 25$.

$\mathbf{x}_0$	E_f	E_G	R_E
$[0,...,0]^\top$	171072	444101	1.57893×10^{-22}
$[3,...,3]^\top$	312888	644091	2.88783×10^{-22}
$[-5,...,-5]^\top$	176624	444101	1.63017×10^{-22}
$[2,-2,...,2,-2,2]^\top$	173472	444101	1.60108×10^{-22}
$[3,-3,...,3,-3,3]^\top$	191402	563646	1.76656×10^{-22}
$[5,-5,...,5,-5,5]^\top$	193297	563646	1.78405×10^{-22}

TABLE VIII
RESULTS OF ALGORITHM 3 - ROSENBROCK'S FUNCTION FOR $n = 25$.

$\mathbf{x}_0$	E_f	E_G	R_E	N_T
$[0,...,0]^\top$	115187	234568	1.06313×10^{-22}	1
$[3,...,3]^\top$	116338	234006	1.07375×10^{-22}	1
$[-5,...,-5]^\top$	111952	211717	1.03327×10^{-22}	1
$[2,-2,...,2,-2,2]^\top$	109246	218561	1.0083×10^{-22}	1
$[3,-3,...,3,-3,3]^\top$	118594	237145	1.09458×10^{-22}	1
$[5,-5,...,5,-5,5]^\top$	122866	231468	1.1340×10^{-22}	1

TABLE IX
COMPARISON OF ALGORITHMS 2 AND 3

Problems	Types	$E_{f,avg}$	$E_{G,avg}$	$R_{E,avg}$
Colville's function	Alg. 2	1679.5	5247.2	0.008635805
	Alg. 3	1143.2	2954.7	0.005878038
Goldstein and Price's function	Alg. 2	22249	151356	0.0013899
	Alg. 3	18369.16667	62131.5	0.001147499
Rosenbrock's function for $n = 5$	Alg. 2	1427.909	4803.636	0.456931
	Alg. 3	1068.364	2489.364	0.341876
Rosenbrock's function for $n = 25$	Alg. 2	203125.8333	517281	1.87477×10^{-22}
	Alg. 3	115697.1667	227910.8333	1.06784×10^{-22}

minimum is $\mathbf{x}^*_{\text{global}} = [1,1,1,1,1]^\top$ with $f(\mathbf{x}^*_{\text{global}}) = 0$. Eleven starting points are used to start Algorithm 2. These are $[-5,-5,-5,-5,-5]^\top$, $[-4,-4,-4,-4,-4]^\top$, $[-3,-3,-3,-3,-3]^\top$, $[-2,-2,-2,-2,-2]^\top$, $[-1,-1,-1,-1,-1]^\top$, $[0,0,0,0,0]^\top$, $[1,1,1,1,1]^\top$, $[2,2,2,2,2]^\top$, $[3,3,3,3,3]^\top$, $[4,4,4,4,4]^\top$, and $[5,5,5,5,5]^\top$. Algorithm 2 was able to determine the global solution from all starting points.

Table VI shows the numerical results of minimizing Rosenbrock's function using Algorithm 3. Note that this algorithm requires far fewer evaluations of both f and $G_{\mu,\rho,\mathbf{x}^*}$ when compared with Algorithm 2 discussed earlier. Similar to Problem 1, the gain in efficiency for Algorithm 3 is offset by reduced reliability, where we have to repeat the algorithm several times for certain starting points, namely $[-1,-1,-1,-1,-1]^\top$, $[0,0,0,0,0]^\top$, $[3,3,3,3,3]^\top$, and $[4,4,4,4,4]^\top$, before a global solution is attained as shown in Table VI. Specifically, when it works, Algorithm 3 succeeds in finding the global solution of the problem with an average $R_E = 0.341876$, compared with $R_E = 0.456931$ obtained by Algorithm 2. In other words, Algorithm 3 is able to minimize Rosenbrock's function much more efficiently than Algorithm 2, with a reduction of 25% in the total number of original function evaluations.

Besides, we also tested both algorithms on a 25-dimensional Rosenbrock's function which has 1.08×10^{26} feasible points and summarized the outcomes in Tables VII and VIII, respectively. Six starting points are used in running the problem, which are $[0,...,0]^\top$, $[3,...,3]^\top$, $[-5,...,-5]^\top$, $[2,-2,...,2,-2,2]^\top$, $[3,-3,...,3,-3,3]^\top$, and $[5,-5,...,5,-5,5]^\top$. The results shows that the proposed Algorithm 3 reduces the original function evaluations significantly by 43.04% compared to the standard algorithm. Interestingly, Algorithm 3 also managed to reach the global solution without repeating the algorithm for each starting point as shown in Table VIII.

A summary of the computational results from using both algorithms to solve Problems 1 to 3 is shown in Table IX. Both discrete filled function algorithms eventually succeeded in finding the global solutions of Colville's, Goldstein and Price's, and Rosenbrock's functions from all starting points, although the modified algorithm required repeated starts for some initial points for Colville's and Rosenbrock's functions in the case of $n = 5$. Clearly, our proposed modification for the discrete filled function algorithm has shown promising results in accelerating convergence to the optimal solution. This is shown through the much lower original function and filled function evaluations required. Future work will include applying the new modification in solving application problems, and keep track a new update in this area such as [14].

REFERENCES

[1] Ali Khater Mohamed, Ali Wagdy Mohamed, Ehab Zaki Elfeky, and Mohamed Saleh. Solving constrained non-linear integer and mixed-integer global optimization problems using enhanced directed differential evolution algorithm. In *Machine Learning Paradigms: Theory and Application*, pages 327–349. Springer, 2019.

[2] R. Ge. A filled function method for finding a global minimizer of a function of several variables. *Mathematical Programming*, 46(1):191–204, 1990.

[3] W. Zhu. An approximate algorithm for nonlinear integer programming. *Applied Mathematics and Computations*, 93(2-3):183–193, 1998.

[4] C. K. Ng, D. Li, and L. S. Zhang. Discrete global descent method for discrete global optimization and nonlinear integer programming. *Journal of Global Optimization*, 37(3):357–379, 2007.

[5] Y. Shang and L. Zhang. Finding discrete global minima with a filled function for integer programming. *European Journal of Operational Research*, 189(1):31–40, 2008.

[6] Y. Yang, Z. Wu, and F. Bai. A filled function method for constrained nonlinear integer programming. *Journal of Industrial and Management Optimization*, 4(2):353–362, 2008.

[7] Hongwei Lin, Yuping Wang, Lei Fan, and Yuelin Gao. A new discrete filled function method for finding global minimizer of the integer programming. *Applied Mathematics and Computation*, 219(9):4371–4378, 2013.

[8] Jinrui Li, Youlin Shang, and Ping Han. Discrete tunnel-filled function method for discrete global optimization problems. In *Proceedings of the Fourth International Forum on Decision Sciences*, pages 873–882. Springer, 2017.

[9] Zhi Guo Feng, Ka Fai Cederic Yiu, and Soon Yi Wu. Design of sparse filters by a discrete filled function technique. *Circuits, System in Signal Processing*, 37:4279–4294, 2018.

[10] S. F. Woon and V. Rehbock. A critical review of discrete filled function methods in solving nonlinear discrete optimization problems. *Applied Mathematics and Computation*, 217:25–41, 2010.

[11] Y. Shang and L. Zhang. A filled function method for finding a global minimizer on global integer optimization. *Journal of Computational and Applied Mathematics*, 181(1):200–210, 2005.

[12] Y. Yang and L. Zhang. A gradually descent method for discrete global optimization. *Journal of Shanghai University (English Edition)*, 11(1):39–44, 2007.

[13] K. Schittkowski. *More test examples for nonlinear programming codes*. Springer-Verlag, New York, 1987.

[14] Juan Pablo Di Mauro and H. Scolnik. An augmented filled function for global nonlinear integer optimization. *TOP*, pages 1–16, 2020.