Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational and Applied Mathematics 220 (2008) 643–660
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

A discrete filled function algorithm for approximate global solutions of max-cut problems☆

Ai-Fan Ling[a,*], Cheng-Xian Xu[a,b], Feng-Min Xu[a]

[a]*Department of Mathematics, Faculty of Science, Xi'an Jiaotong University, Xi'an 710049, PR China*
[b]*SKLMSE Lab., Xi'an Jiaotong University, PR China*

Received 3 January 2007; received in revised form 24 July 2007

Abstract

A discrete filled function algorithm is proposed for approximate global solutions of max-cut problems. A new discrete filled function is defined for max-cut problems and the properties of the filled function are studied. Unlike general filled function methods, using the characteristic of max-cut problems, the parameters in proposed filled function need not be adjusted. This greatly increases the efficiency of the filled function method. By combining a procedure that randomly generates initial points for minimization of the filled function, the proposed algorithm can greatly reduce the calculation cost and be applied to large scale max-cut problems. Numerical results on different sizes and densities test problems indicate that the proposed algorithm is efficient and stable to get approximate global solutions of max-cut problems.

Keywords: Combinatorial optimization; Global optimization; Filled function; Max-cut problem; Neighborhood local search

1. Introduction

Filled function methods introduced by Ge in 1990 are a class of global optimization methods. Consider the following optimization problem:

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in \Omega \subset \mathbb{R}^n, \end{aligned} \tag{1.1}$$

where $f: \mathbb{R}^n \to \mathbb{R}$ is a real valued continuous function. Let $\mathbf{x}_1^*$ be a local minimizer of problem (1.1), the basic idea of the filled function method is to construct an auxiliary function, called filled function, at point $\mathbf{x}_1^*$, that can be further minimized to get a point, say $\bar{\mathbf{x}}$, in a basin (see Definition 5 below) of $f(\mathbf{x})$ lower than the basin containing $\mathbf{x}_1^*$ of $f(\mathbf{x})$ when $\mathbf{x}_1^*$ is not a global minimizer. Then the minimization of $f(\mathbf{x})$ is restarted from the point $\bar{\mathbf{x}}$. Repeat the process until a global minimizer of $f(\mathbf{x})$ is obtained.

☆ This work is supported by National Natural Science Foundations of China 10671152.
* Corresponding author.
E-mail addresses: aifanling@yahoo.com.cn (A.-F. Ling), mxxu@mail.xjtu.edu.cn (C.-X. Xu), fengminxu@mail.xjtu.edu.cn (F.-M. Xu).

0377-0427/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2007.09.012

The first filled function proposed in [13] has the form

$$P(\mathbf{x};\mathbf{x}^*;r,\rho)=\frac{1}{r+f(\mathbf{x})}\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}^*\|_2^2}{\rho^2}\right), \tag{1.2}$$

where $\mathbf{x}^*$ is a minimizer of the function f, r and ρ are two adjustable parameters. An unfavorable property of the filled function (1.2) is that the existence of term $\exp(-\|\mathbf{x}-\mathbf{x}^*\|_2^2/\rho^2)$ makes changes in $P(\mathbf{x};\mathbf{x}^*;r,\rho)$ and $\nabla P(\mathbf{x};\mathbf{x}^*;r,\rho)$ indistinguishable when $\|\mathbf{x}-\mathbf{x}^*\|_2^2$ is large. Liu in [21] proposed the following filled function:

$$L(\mathbf{x};\mathbf{x}^*;a)=\frac{1}{\ln[1+f(\mathbf{x})-f(\mathbf{x}^*)]}-a\|\mathbf{x}-\mathbf{x}^*\|_2^2$$

to improve the property of filled function (1.2), where a is a parameter. Xu et al. in [31] proposed a class of filled functions to overcome the disadvantage

$$U(\mathbf{x};\mathbf{x}^*;A,\gamma)=-\eta(f(\mathbf{x})-f(\mathbf{x}^*))-A\psi(\|\mathbf{x}-\mathbf{x}^*\|_2^\gamma),$$

where the functions $\eta(\cdot)$, $\psi(\cdot)$ have some desired properties and A,γ are parameters. Recently, Zhang et al. in [32] proposed a new two-parameter filled function

$$p(\mathbf{x};\mathbf{x}^*;\rho,\mu)=\begin{cases}\mu[f(\mathbf{x})-f(\mathbf{x}^*)]^2-\rho\|\mathbf{x}-\mathbf{x}^*\|_2^2 & \text{when } f(\mathbf{x})\geqslant f(\mathbf{x}^*),\\ f(\mathbf{x}^*)-f(\mathbf{x})-\rho\|\mathbf{x}-\mathbf{x}^*\|_2^2 & \text{when } f(\mathbf{x})\leqslant f(\mathbf{x}^*).\end{cases} \tag{1.3}$$

Ge and Huang [14] employed the filled function method to solve nonlinear integer programming problems. Ng et al. [23] dealt with discrete optimization problems using the filled function (1.3). More results on filled function methods for discrete optimization problems can be found in Zhu [33], He et al. [18], Shang and Zhang [26] and Gu and Wu [17]. However, to the best of our knowledge, there is very few attempts for max-cut problems or other combinatorial optimization problems by using the filled function methods.

Max-cut problems are a kind of discrete optimization problems. Given a graph $G(V;E)$, the problem is to find a partition of the node set $S_1\subset V$ and $S_2=V\backslash S_1$, such that the sum of the weights on the edges connecting the two parts is maximized, where V and E are the sets of nodes and edges in the graph. Let $W=(w_{ij})_{n\times n}$ be the symmetric weighted adjacency matrix of a given graph $G(V,E)$, with $w_{ij}\neq 0$, for $(i,j)\in E$ and $w_{ij}=0$ for $(i,j)\notin E$, the max-cut problem can be expressed as the following discrete quadratic optimization problem:

$$(MC):\quad \min\quad f(\mathbf{x})=\mathbf{x}^{\mathrm{T}}W\mathbf{x}$$
$$\text{s.t.}\quad \mathbf{x}\in\mathbb{S},$$

where $\mathbb{S}=\{-1,1\}^n$ and $\mathbf{x}=(x_1,\ldots,x_n)^{\mathrm{T}}$. $\mathbf{x}\in\{-1,1\}^n$ means that x_i takes either -1 or 1 for all $i=1,2,\ldots,n$.

The max-cut problem has long been known to be NP-complete [20], even for any un-weighted graphs [12], and has applications in circuit layout design and statistical physics [3]. Approximate algorithms, such as ρ-approximation algorithm [16], heuristic algorithms [9] and continuous algorithms [27,30] have been proposed to solve max-cut problems. Based on the following semi-definite programming (SDP) relaxation,

$$(SDP):\quad \max\quad L\cdot X$$
$$\text{s.t.}\quad \operatorname{diag}(X)=\mathbf{e},$$
$$X\succcurlyeq 0,$$

Goemans and Williamson in [16] proposed a 0.878-approximation randomized algorithm for nonnegative weighted graphs, where $\mathbf{e}\in\mathbb{R}^n$ is the column vector with all ones and $L=[\operatorname{Diag}(W\mathbf{e})-W]/4$, $\operatorname{Diag}(W\mathbf{e})$ is the diagonal matrix with elements of the vector $W\mathbf{e}$ being diagonal entries and $\operatorname{diag}(X)=(X_{11},\ldots,X_{nn})^{\mathrm{T}}$, $X_{ii}, i=1,2,\ldots,n$ are the diagonal entries of the matrix X. Bertsimas-Ye in [6] and Zwick in [34] also proposed random algorithms for nonnegative weighted graphs with the same performance ratio.

Burer et al. in [9] proposed a rank-2 relaxation to max-cut problems and developed a continuous optimization heuristic to solve max-cut problems. Festa et al. in [11] designed, implemented and tested several pure and hybrid heuristics, such as greedy randomized adaptive search procedure (GRASP), variable neighborhood search (VNS) and a path-relinking (PR) intensification heuristic for max-cut problems and obtained some satisfactory results on some well-known G-set graphs. Alperin and Nowak in [2] presented a smoothing heuristic for max-cut problems. The heuristic is based on a parametric optimization problem defined as a convex combination between the original problem and its Lagrangian relaxation. Starting from the Lagrangian relaxation, a path following method is proposed to obtain good solutions while gradually transforming the relaxed problem into the original problem formulated with an exact penalty function.

Xu et al. in [30,27] proposed continuous optimization methods to solve max-cut problems, in which max-cut problems were relaxed into a nonlinear continuous optimization problem with convex constraints:

$$\begin{aligned} \max \quad & \mathbf{x}^{\mathrm{T}} L \mathbf{x} \\ \text{s.t.} \quad & B(x_i, \sqrt{n}) \leqslant 0, \quad i = 1, \ldots, n, \\ & \|\mathbf{x}\|_2^2 \leqslant n, \end{aligned}$$

where $B(x_i, \sqrt{n})$ is a B-function (see [30]). An obvious advantage of the continuous method is that it greatly reduces the CPU-time via without using linear search in each iteration. However, the solution obtained by the continuous methods cannot be guaranteed as a global solution, which motivates us to study global optimization method to deal with max-cut problems.

The main contribution of this paper is to propose a discrete filled function method to get approximate global solutions of max-cut problems. A discrete filled function is applied to solve max-cut problems for different size and dense graphs concluding negative weight. Only at most $(n - K)(K \in \mathbb{Z}, 1 \leqslant K < n)$ inner products of two n-dimensional vectors are calculated in each iteration, and the parameters of the filled function need not to be adjusted in the process of implementing the algorithm. By adding a subroutine for randomly generating initial points to minimizing the filled function, the proposed algorithm can greatly reduce the calculation cost, and can be applied to large scale max-cut problems. Numerical experiments and comparisons on some randomly generated max-cut problems and on some well-known large scale max-cut problems are made to show the efficiency of the proposed algorithm.

The remainder of the paper is arranged as follows. In Section 2, some definitions and preliminaries are recalled. The new discrete filled function for problem (MC) is presented in Section 3, and the properties of the filled function are analyzed. In Section 4, details of the proposed filled function algorithm is stated. Numerical experiments and comparisons are reported in Section 5.

Throughout the paper, if without special statement, we adopt the following convention. $\mathbb{R}$, $\mathbb{Z}$ and $\mathbb{R}^n$ denote real numbers set, integer numbers set and the space of real n-dimensional column vectors, respectively. Let $\mathbf{x} \in \mathbb{R}^n$, the l_p-norm of $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p} (1 \leqslant p < \infty)$.

2. Definition and preliminaries

For convenience, in this section, we will give some definitions and lemmas without proofs.

Definition 1. A sequence $\{\mathbf{x}^{(i)}\}_{i=-1}^u$ is called a discrete path in the set $\mathbb{S}$ between two points $\mathbf{x}^*$, $\mathbf{x}^{**}$ in $\mathbb{S}$, where $\mathbf{x}^{(-1)} = x^*$, $\mathbf{x}^{(u)} = \mathbf{x}^{**}$, $\mathbf{x}^{(i)} \in \mathbb{S}$ for all i, if $\mathbf{x}^{(i)} \neq \mathbf{x}^{(j)}$, for all $i \neq j$; and $\|\mathbf{x}^{(i)} - \mathbf{x}^{(i-1)}\|_p = 2$, for all $i = 0, 1, 2, \ldots, u$. In addition, if $\|\mathbf{x}^{(i)} - \mathbf{x}^*\|_p < \|\mathbf{x}^{(i+1)} - \mathbf{x}^*\|_p$, or $\|\mathbf{x}^{(i)} - \mathbf{x}^*\|_p > \|\mathbf{x}^{(i+1)} - \mathbf{x}^*\|_p$ holds for all i, then the sequence is called a strict discrete path in $\mathbb{S}$. If a (strict) discrete path exists for given points $\mathbf{x}^* \in \mathbb{S}$, $\mathbf{x}^{**} \in \mathbb{S}$, then $\mathbf{x}^*$ and $\mathbf{x}^{**}$ are said to be (strict) path-wise connected in $\mathbb{S}$.

For simplicity, we call a (strict) discrete path as a (strict) path. From the definition of the path and the property of the set $\mathbb{S}$, it is clear that any two distinct points $\mathbf{x}, \mathbf{y} \in \mathbb{S}$ are (strict) path-wise connected, and hence $\mathbb{S}$ is a (strict) path-wise connected domain.

Definition 2. For any $\mathbf{x} \in \mathbb{S}$, and any positive integer K, $1 \leqslant K \leqslant n$, the K-neighborhood of $\mathbf{x}$ under the l_p-norm is defined by

$$N(\mathbf{x}, K) = \{\mathbf{y} \in \mathbb{S} : \|\mathbf{y} - \mathbf{x}\|_p \leqslant 2 \cdot K^{1/p}\}.$$

In particular, if $K = 1$, we write the 1-neighborhood of $\mathbf{x}$, $N(\mathbf{x}, 1)$, as $N(\mathbf{x})$. The boundary of the K-neighborhood $N(\mathbf{x}, K)$ is defined by $\partial N(\mathbf{x}, K) = \{\mathbf{y} \in \mathbb{S} : \|\mathbf{y} - \mathbf{x}\|_p = 2 \cdot K^{1/p}\}$. It is known that $|\partial N(\mathbf{x}, K)| = C_n^K$ and each point in $\partial N(\mathbf{x}, K)$ differs from point $\mathbf{x}$ in only K elements. The following lemma is obvious from Definition 2.

Lemma 2.1. *For every $K \in \mathbb{Z}$, $1 \leqslant K \leqslant n$, and any $\mathbf{x} \in \mathbb{S}$, the number of points in $N(\mathbf{x}, K)$ is $1 + \sum_{i=1}^{K} C_n^i$. Especially, when $K = 1$, $|N(\mathbf{x})| = n + 1$.*

Definition 3. A point $\mathbf{x}^* \in \mathbb{S}$ is called a local discrete minimizer of the function f over $\mathbb{S}$, if $f(\mathbf{x}^*) \leqslant f(\mathbf{x})$, for all $\mathbf{x} \in N(\mathbf{x}^*)$. Furthermore, if $f(\mathbf{x}^*) \leqslant f(\mathbf{x})$, for all $\mathbf{x} \in \mathbb{S}$, then $\mathbf{x}^*$ is called a global discrete minimizer of f over $\mathbb{S}$.

Let $\mathbf{e}_i \in \mathbb{R}^n$ be the ith unit directional vector, and define a set of directions $\mathscr{D} = \{\mathbf{d} : \mathbf{d} = \pm 2 \cdot \mathbf{e}_i, i = 1, \ldots, n\}$, then for any $\mathbf{x} \in \mathbb{S}$, there exists a direction $\mathbf{d} \in \mathscr{D}$, such that $\mathbf{x} + \mathbf{d} \in N(\mathbf{x}) \subset \mathbb{S}$.

Definition 4. For any $\mathbf{x} \in \mathbb{S}$, a direction $\mathbf{d} \in \mathscr{D}$ is called a descent direction of the function f at $\mathbf{x}$ over $\mathbb{S}$, if $\mathbf{x} + \mathbf{d} \in \mathbb{S}$ and $f(\mathbf{x} + \mathbf{d}) < f(\mathbf{x})$. In addition, we call $\mathbf{d}^* \in \mathscr{D}_\mathbf{x}$ the discrete steepest descent direction of the function f at $\mathbf{x}$ over $\mathbb{S}$, if $f(\mathbf{x} + \mathbf{d}^*) \leqslant f(\mathbf{x} + \mathbf{d})$ for all $\mathbf{d} \in \mathscr{D}_\mathbf{x}$, where $\mathscr{D}_\mathbf{x} = \{\mathbf{d} \in \mathscr{D} : \mathbf{x} + \mathbf{d} \in \mathbb{S}, f(\mathbf{x} + \mathbf{d}) < f(\mathbf{x})\}$.

Definition 5. Let $\mathbf{x}^*$ be a local minimizer of the function f over $\mathbb{S}$, B^* is said to be a discrete basin of the function f at $\mathbf{x}^*$, if $B^* \subset \mathbb{S}$ is a path-wise connected domain which contains $\mathbf{x}^*$ and in which the discrete steepest descent trajectory from any point in B^* converges to $\mathbf{x}^*$, but outside which the discrete steepest descent trajectory from any point in $\mathbb{S} \backslash B^*$ does not converge to $\mathbf{x}^*$.

Definition 6 (*Xu et al.* [31]). Let $\mathbf{x}^*$ and $\mathbf{x}^{**}$ be two distinct local minimizers of the function f over $\mathbb{S}$, the discrete basin B^{**} of f at $\mathbf{x}^{**}$ is said to be lower (or higher) than the discrete basin B^* of f at $\mathbf{x}^*$, if $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$ (or $f(\mathbf{x}^{**}) > f(\mathbf{x}^*)$).

Definition 7 (*Ng et al.* [23]). Let $\mathbf{x}^*$ be a discrete minimizer of the function f over $\mathbb{S}$, and B^* be the discrete basin of f at $\mathbf{x}^*$. A function $H : \mathbb{S} \to \mathbb{R}$ is said to be a discrete filled function of f at $\mathbf{x}^*$, if it satisfies the following conditions:

1. $\mathbf{x}^*$ is a strict discrete local maximizer of H over $\mathbb{S}$.
2. H has no discrete local minimizers in B^* and in any discrete basin of f higher than B^*.
3. If f has a discrete basin, B^{**} say, lower than B^*, then there is a point $\bar{\mathbf{x}} \in B^{**}$ that minimizes H on the path $\{\mathbf{x}^*, \ldots, \bar{\mathbf{x}}, \ldots, \mathbf{x}^{**}\}$ in $\mathbb{S}$, where $\mathbf{x}^{**}$, $\mathbf{x}^*$ are minimizers of f in B^{**} and B^*, respectively.

3. A discrete filled function and its properties

Let $\mathbf{x}^*$ be a local minimizer of the function f in problem (MC); we modify the filled function (1.3) and define a function with simpler expression as

$$H_p(\mathbf{x}; \mathbf{x}^*; \alpha, \beta) = \begin{cases} \alpha[f(\mathbf{x}) - f(\mathbf{x}^*)] - \beta\|\mathbf{x} - \mathbf{x}^*\|_p, & f(\mathbf{x}) \geqslant f(\mathbf{x}^*), \\ [f(\mathbf{x}^*) - f(\mathbf{x})] - \beta\|\mathbf{x} - \mathbf{x}^*\|_p, & f(\mathbf{x}) \leqslant f(\mathbf{x}^*), \end{cases} \tag{3.1}$$

where $p(1 \leqslant p < \infty)$ is a constant, and α, β are two adjustable parameters. For simplicity, in the remainder of this paper, we only consider the case of $p = 1$, and denote $H_1(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ as $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$, but it is not hard to extend all results of this paper to the case of $1 < p < \infty$. Since the specialities of the quadratic function f and the set $\mathbb{S}$ in max-cut problems, we can simplify the proofs to show that $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ is a filled function of f at $\mathbf{x}^*$, and assign a more efficient algorithm for global minimization of max-cut problems. The remainder of this section will be used to show that the function $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ is a discrete filled function of the function f at point $\mathbf{x}^*$. The following lemma plays an important role in the analysis of this section.

Lemma 3.1. *For any given $\mathbf{x} \in \mathbb{S}$ and for any $\mathbf{y} \in N(\mathbf{x})$, we have*

$$|f(\mathbf{y}) - f(\mathbf{x})| \leqslant 4qM, \tag{3.2}$$

where $q = \max\{q_i : i = 1, \dots, n\} \leqslant n$ *and* $M = \max\{\|\mathbf{w}_i\|_1 : i = 1, \dots, n\}$, $\mathbf{w}_i$ $(i = 1, \dots, n)$ *are the* i*th column of the weighted adjacency matrix* W *of a given graph, and* $q_i\,(i = 1, \dots, n)$ *are the number of nonzero elements in vector* $\mathbf{w}_i$ $(i = 1, \dots, n)$.

Proof. Let $\mathbf{x}^l \in N(\mathbf{x})$ $(l = 1, 2, \dots, n)$, where index l expresses that only the lth element of $\mathbf{x}^l$ differs from the lth element of $\mathbf{x}$, i.e.,

$$x_i^l = \begin{cases} x_i, & i = 1, 2, \dots, n, \quad i \neq l, \\ -x_l, & i = l. \end{cases}$$

Since $w_{ii} = 0$, for $i = 1, \dots, n$, we have

$$\begin{aligned} |f(\mathbf{x}^l) - f(\mathbf{x})| &= |(\mathbf{x}^l)^{\mathrm{T}} W \mathbf{x}^l - \mathbf{x}^{\mathrm{T}} W \mathbf{x}| \\ &= \left| \sum_{i \neq l, j \neq l}^{n} x_i w_{ij} x_j - 2 \sum_{i \neq l}^{n} x_i w_{il} x_l - \sum_{i,j}^{n} x_i w_{ij} x_j \right| \\ &= 4 \left| \sum_{i=1}^{n} x_i w_{il} x_l \right| \\ &= 4 |x_l \cdot \mathbf{x}^{\mathrm{T}} \mathbf{w}_l| \\ &= 4 |\mathbf{x}^{\mathrm{T}} \mathbf{w}_l| \leqslant 4 q_l \|\mathbf{w}_l\|_1 \\ &\leqslant 4qM, \quad l = 1, 2, \dots, n. \end{aligned} \tag{3.3}$$

This completes the proof. □

Theorem 3.2. *Let* $\mathbf{x}^*$ *be a local minimizer of the function* f *on* $\mathbb{S}$. *If* $\beta > 0$, *and* $0 \leqslant \alpha < \beta/(2qM)$, *then the function* $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ *satisfies the first condition of Definition 7, i.e.,* $\mathbf{x}^*$ *is a strict local maximizer of* $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ *on* $\mathbb{S}$. *Furthermore, if* $\mathbf{x}^*$ *is a global minimizer of* f *on* $\mathbb{S}$, *then* $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta) < 0$ *for all* $\mathbf{x} \in \mathbb{S} \backslash \{\mathbf{x}^*\}$.

Proof. Assume that $\mathbf{x}^*$ is a local minimizer of f on $\mathbb{S}$. Since $f(\mathbf{x}) \geqslant f(\mathbf{x}^*)$ and $\|\mathbf{x} - \mathbf{x}^*\|_1 = 2$ for any $\mathbf{x} \in N(\mathbf{x}^*) \backslash \{\mathbf{x}^*\}$. It follows from Lemma 3.1 that when $0 \leqslant \alpha < \beta/(2qM)$, we have

$$\begin{aligned} H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta) &= \alpha[f(\mathbf{x}) - f(\mathbf{x}^*)] - \beta \|\mathbf{x} - \mathbf{x}^*\|_1 \\ &\leqslant 4\alpha q M - 2\beta < 0 = H(\mathbf{x}^*; \mathbf{x}^*; \alpha, \beta). \end{aligned}$$

Hence, $\mathbf{x}^*$ is a strict local maximizer of $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ on $\mathbb{S}$.

If $\mathbf{x}^*$ is a global minimizer of f on $\mathbb{S}$, then for any $\mathbf{x} \in \mathbb{S} \backslash \{\mathbf{x}^*\}$, there exists a integer $K (1 \leqslant K \leqslant n)$, such that $\mathbf{x} \in \partial N(\mathbf{x}^*, K)$ that implies $\|\mathbf{x} - \mathbf{x}^*\|_1 = 2K$. Hence, there must exist at least a path $\{\mathbf{x}_j\}_{j=0}^{K}$ from $\mathbf{x}^*$ to $\mathbf{x}$ satisfying $\mathbf{x}^* = \mathbf{x}_0$, $\mathbf{x} = \mathbf{x}_K$. Since

$$\begin{aligned} |f(\mathbf{x}) - f(\mathbf{x}^*)| &= |f(\mathbf{x}_K) - f(\mathbf{x}_{K-1}) + f(\mathbf{x}_{K-1}) - f(\mathbf{x}_{K-2}) + \cdots + f(\mathbf{x}_1) - f(\mathbf{x}_0)| \\ &\leqslant \sum_{i=1}^{K} |f(\mathbf{x}_i) - f(\mathbf{x}_{i-1})|, \end{aligned}$$

and $\mathbf{x}_i \in N(\mathbf{x}_{i-1})$. It follows from Lemma 4.1 that

$$|f(\mathbf{x}) - f(\mathbf{x}^*)| \leqslant \sum_{i=1}^{K} 4qM = 4KqM.$$

Thus,

$$\begin{aligned} H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta) &= \alpha[f(\mathbf{x}) - f(\mathbf{x}^*)] - \beta \|\mathbf{x} - \mathbf{x}^*\|_1 \\ &\leqslant 4K\alpha q M - 2K\beta < 0 = H(\mathbf{x}^*; \mathbf{x}^*; \alpha, \beta), \end{aligned}$$

when $\beta > 0$, and $0 \leqslant \alpha < \beta/(2qM)$. □

Lemma 3.3. *Let* $\mathbf{x}^*$ *be a local minimizer of the function f on* $\mathbb{S}$, $\overline{\mathbf{x}} \in \mathbb{S}$ *and* $f(\overline{\mathbf{x}}) \geqslant f(\mathbf{x}^*)$. *If* $\beta > 0$ *and* $0 \leqslant \alpha \leqslant \beta/(2qM)$, *and if there exists direction* $\overline{\mathbf{d}} \in \mathscr{D}$, *such that* $\overline{\mathbf{x}} + \overline{\mathbf{d}} \in \mathbb{S}$, $f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) \geqslant f(\mathbf{x}^*)$ *and* $\|\overline{\mathbf{x}} + \overline{\mathbf{d}} - \mathbf{x}^*\|_1 > \|\overline{\mathbf{x}} - \mathbf{x}^*\|_1$, *then we have*

$$H(\overline{\mathbf{x}} + \overline{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) \leqslant H(\overline{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^*; \mathbf{x}^*; \alpha, \beta) = 0. \tag{3.4}$$

Proof. Denote $\mathscr{D}_0 = \{\overline{\mathbf{d}} \in \mathscr{D} : \overline{\mathbf{x}} + \overline{\mathbf{d}} \in \mathbb{S},\ f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) \geqslant f(\mathbf{x}^*),\ \|\overline{\mathbf{x}} + \overline{\mathbf{d}} - \mathbf{x}^*\|_1 > \|\overline{\mathbf{x}} - \mathbf{x}^*\|_1\}$. If $\mathscr{D}_0$ is nonempty, then $f(\overline{\mathbf{x}}) \geqslant f(\mathbf{x}^*)$, $f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) \geqslant f(\mathbf{x}^*)$ and $\|\overline{\mathbf{x}} + \overline{\mathbf{d}} - \mathbf{x}^*\|_1 - \|\overline{\mathbf{x}} - \mathbf{x}^*\|_1 = 2$ for each $\overline{\mathbf{d}} \in \mathscr{D}_0$. It follows from the definition of the function H that

$$H(\overline{\mathbf{x}} + \overline{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) - H(\overline{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) = \alpha[f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) - f(\overline{\mathbf{x}})] - 2\beta. \tag{3.5}$$

If $f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) \leqslant f(\overline{\mathbf{x}})$, then $H(\overline{\mathbf{x}} + \overline{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) < H(\overline{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta)$ when $\beta > 0$. If $f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) \geqslant f(\overline{\mathbf{x}})$, it follows from Lemma 3.1 that the right-hand side of (3.5) is not greater than $4\alpha qM - 2\beta$, i.e.,

$$H(\overline{\mathbf{x}} + \overline{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) - H(\overline{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) \leqslant 4\alpha qM - 2\beta.$$

Hence, when $0 \leqslant \alpha \leqslant \beta/(2qM)$,

$$H(\overline{\mathbf{x}} + \overline{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) \leqslant H(\overline{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta)$$

holds. The second inequality in (3.4) directly follows from Theorem 3.2. □

Lemma 3.3 indicates that for any point $\overline{\mathbf{x}} \in \mathbb{S}$, especially, when $\overline{\mathbf{x}}$ lies either in the basin B^* or basin higher than B^*, if $\mathscr{D}_0$ is nonempty, then $\overline{\mathbf{x}}$ is not a local minimizer of the function H. That is, we have the following result.

Theorem 3.4. *Let* $\mathbf{x}^*$ *be a local minimizer of the function f on* $\mathbb{S}$, *and* B^* *be the discrete basin of f at* $\mathbf{x}^*$. *If* $\beta > 0$, $0 \leqslant \alpha < \beta/(2qM)$, *then* $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ *has no any local minimizer in* B^* *and in any basin higher than* B^*.

Proof. Let U^* denote the union of all basins of f higher than B^*, it is clear from Definition 6 that $f(\mathbf{x}) > f(\mathbf{x}^*)$ holds for any point $\mathbf{x} \in U^*$. Note that $-\mathbf{x}^* \notin U^*$ via $f(-\mathbf{x}^*) = f(\mathbf{x}^*)$ and $-\mathbf{x}^* \notin B^*$ via the fact that $\mathbf{x}^*$ oneself is a minimizer of f. Hence for any $\overline{\mathbf{x}} \in B^*$ or $\overline{\mathbf{x}} \in U^*$, it follows that $\overline{\mathbf{x}} \neq -\mathbf{x}^*$. Thus, there exists at least a direction $\overline{\mathbf{d}} \in \mathscr{D}$, such that $\overline{\mathbf{x}} + \overline{\mathbf{d}} \in \mathbb{S}$ and $\|\overline{\mathbf{x}} + \overline{\mathbf{d}} - \mathbf{x}^*\|_1 > \|\overline{\mathbf{x}} - \mathbf{x}^*\|_1$. If $f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) < f(\mathbf{x}^*)$, then we can obtain a point, say $\mathbf{x}^{**}$, satisfying $f(\mathbf{x}^{**}) < f(\mathbf{x}^*)$ by minimizing the function f from $\overline{\mathbf{x}}$. It means $\overline{\mathbf{x}}$ is in a basin of f lower than the basin B^* that contradicts $\overline{\mathbf{x}} \in B^*$ or $\overline{\mathbf{x}} \in U^*$. Thus the set $\mathscr{D}_0$ is nonempty. From Lemma 3.3, $\overline{\mathbf{x}}$ is not a local minimizer of the function $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$. □

Theorem 3.4 indicates that if $\beta > 0$ and $0 \leqslant \alpha < \beta/(2qM)$, then $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ satisfies the second condition of Definition 7. From Lemma 3.3 and Theorem 3.4, we can obtain the following conclusions. When $\beta > 0$ and $0 \leqslant \alpha < \beta/(2qM)$,

(i) for any $\mathbf{x} \in B^*$, $\mathbf{x} \neq \mathbf{x}^*$, if $\mathbf{d}$ satisfies $\|\mathbf{x} + \mathbf{d} - \mathbf{x}^*\|_1 > \|\mathbf{x} - \mathbf{x}^*\|_1$ and $f(\mathbf{x} + \mathbf{d}) \geqslant f(\mathbf{x}^*)$, then $\mathbf{d}$ is a descent direction of H at $\mathbf{x}$,
(ii) for any $\mathbf{x} \in U^*$, if $\mathbf{d}$ satisfies $\|\mathbf{x} + \mathbf{d} - \mathbf{x}^*\|_1 > \|\mathbf{x} - \mathbf{x}^*\|_1$ and $f(\mathbf{x} + \mathbf{d}) \geqslant f(\mathbf{x}^*)$, then $\mathbf{d}$ is a descent direction of H at $\mathbf{x}$.

Theorem 3.5. *Let* $f(\mathbf{x}^*) > f(\mathbf{x}^{**})$, *and* B^*, B^{**} *be basins of the function f at* $\mathbf{x}^*$, $\mathbf{x}^{**}$ *in* $\mathbb{S}$, *respectively. Assume that* $\overline{\mathbf{x}} \in B^{**}$, *and* $\overline{\mathbf{x}} + \overline{\mathbf{d}} \in B^{**}$, *where* $\overline{\mathbf{d}}$ *is a descent direction of f at* $\overline{\mathbf{x}}$ *in* $\mathbb{S}$ *and satisfies*

$$f(\overline{\mathbf{x}} + \overline{\mathbf{d}}) < f(\mathbf{x}^*) \leqslant f(\overline{\mathbf{x}}). \tag{3.6}$$

Suppose that there is a strict path $\{\mathbf{x}^{(i)}\}_{i=-1}^{u}$ *in* $\mathbb{S}$ *between* $\mathbf{x}^*(=\mathbf{x}^{(-1)})$ *and* $\overline{\mathbf{x}}(=\mathbf{x}^{(u)})$, *such that*

$$f(\mathbf{x}^{(i)}) \geqslant f(\mathbf{x}^*), \tag{3.7}$$

for all $i = 0, 1, \ldots, u$. *If* $0 \leqslant \alpha < \beta/(2qM)$, *and* $0 < \beta < \beta_0$, *then* $\bar{\mathbf{x}}$ *is a minimizer of the function* $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ *on the path* $\{\mathbf{x}^{(i)}\}_{i=-1}^{u+v}$ *in* $\mathbb{S}$, *where* $\mathbf{x}^*(=\mathbf{x}^{(-1)})$, $\bar{\mathbf{x}} + \bar{\mathbf{d}} = \mathbf{x}^{(u+1)}$, $\mathbf{x}^{**}(=\mathbf{x}^{(u+v)})$ *and*

$$\beta_0 = \frac{2qM[f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}{f(\bar{\mathbf{x}}) - f(\mathbf{x}^*) + 4qM} > 0. \tag{3.8}$$

Proof. It is sufficient to show $H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u+1)}; \mathbf{x}^*; \alpha, \beta) = H(\bar{\mathbf{x}}+\bar{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta)$, and $H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u-1)}; \mathbf{x}^*; \alpha, \beta)$ hold on the path. Since $\mathbf{x}^{(0)}$ and $\mathbf{x}^*$ are adjacent points in the (strict) path $\{\mathbf{x}^{(i)}\}_{i=-1}^{u}$, $\mathbf{x}^{(0)} \in B^*$ and $H(\mathbf{x}^{(0)}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^*; \mathbf{x}^*; \alpha, \beta) = 0$ holds from Theorem 3.2 when $\beta > 0$, $0 \leqslant \alpha < \beta/(2qM)$.

Since $\{\mathbf{x}^{(i)}\}_{i=-1}^{u}$ is a strict path starting from $\mathbf{x}^*$, assume that $\|\mathbf{x}^{(i-1)} - \mathbf{x}^*\|_1 < \|\mathbf{x}^{(i)} - \mathbf{x}^*\|_1$ holds, for all $i = 0, 1, \ldots, u$. In view of (3.7) and Lemma 3.3, for all the points $\{\mathbf{x}^{(i)}\}$, $i = -1, 0, \ldots, u$ on the strict path, we have

$$H(\mathbf{x}^{(i)}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(i-1)}; \mathbf{x}^*; \alpha, \beta) < 0, \quad i = 1, \ldots, u.$$

Especially,

$$H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u-1)}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u-j)}; \mathbf{x}^*; \alpha, \beta), \quad j = 2, \ldots, u+1. \tag{3.9}$$

On the other hand, since $\bar{\mathbf{d}}$ satisfies (3.6),

$$\begin{aligned} &H(\bar{\mathbf{x}} + \bar{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta) - H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) \\ &\quad = f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}}) - \alpha[f(\bar{\mathbf{x}}) - f(\mathbf{x}^*)] - \beta\{\|\bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}^*\|_1 - \|\bar{\mathbf{x}} - \mathbf{x}^*\|_1\} \\ &\quad > f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}}) - \frac{\beta}{2qM}[f(\bar{\mathbf{x}}) - f(\mathbf{x}^*)] - 2\beta \end{aligned} \tag{3.10}$$

holds when $0 \leqslant \alpha < \beta/(2qM)$.

It can be verified that the right-hand side of (3.10) is nonnegative when

$$\beta \leqslant \frac{2qM[f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}{f(\bar{\mathbf{x}}) - f(\mathbf{x}^*) + 4qM} = \beta_0,$$

that is, when $0 < \beta < \beta_0$, we have

$$H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\bar{\mathbf{x}} + \bar{\mathbf{d}}; \mathbf{x}^*; \alpha, \beta). \tag{3.11}$$

Therefore, $\bar{\mathbf{x}}$ minimizes $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ on the path $\{\mathbf{x}^{(i)}\}_{i=-1}^{u+1}$ in $\mathbb{S}$ between $\mathbf{x}^*$ and $\mathbf{x}^{(u+1)} = \bar{\mathbf{x}} + \bar{\mathbf{d}}$. If $\bar{\mathbf{x}} + \bar{\mathbf{d}} = \mathbf{x}^{**}$, then the conclusion of the theorem holds. If $\bar{\mathbf{x}} + \bar{\mathbf{d}} \neq \mathbf{x}^{**}$, since $\bar{\mathbf{x}} + \bar{\mathbf{d}} \in B^{**}$, then there exists a steepest descent trajectory $\{\mathbf{x}^{(i)}\}_{i=u+1}^{u+v}$ of f in $\mathbb{S}$ from $\bar{\mathbf{x}} + \bar{\mathbf{d}}$ to $\mathbf{x}^{**} = \mathbf{x}^{(u+v)}$. Hence $\bar{\mathbf{x}}$ is a minimizer of $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ on the path $\{\mathbf{x}^{(i)}\}_{i=-1}^{u+v}$. □

In view of Theorem 3.5, when $\bar{\mathbf{x}} + \bar{\mathbf{d}} \neq \mathbf{x}^{**}$, the path $\{\mathbf{x}^{(u+j)}\}_{j=1}^{v}$ is a steepest descent trajectory of the function f from $\mathbf{x}^{(u+1)}$, and hence

$$f(\mathbf{x}^*) > f(\mathbf{x}^{(u+1)}) > f(\mathbf{x}^{(u+2)}) > \cdots > f(\mathbf{x}^{(u+v)}) = f(\mathbf{x}^{**}), \tag{3.12}$$

$$\mathbf{x}^{(u+j+1)} \in N(\mathbf{x}^{(u+j)}), \quad j = 1, 2, \ldots, v-1. \tag{3.13}$$

Let

$$\begin{aligned} \beta(j) &= f(\mathbf{x}^{(u+j)}) - f(\mathbf{x}^{(u+j+1)}) \\ &= 4\sum_{i=1}^{n} x_i^{(u+j)} w_{il_j} x_{l_j}^{(u+j)} > 0, \quad j = 1, \ldots, v-1, \end{aligned}$$

where $x_i^{(u+j)}$ is the ith element of the vector $\mathbf{x}^{(u+j)}$, and l_j is defined by

$$l_j = \arg\min\left\{-4\sum_{i=1}^{n} x_i^{(u+j)} w_{il} x_l^{(u+j)} : l = 1, \ldots, n\right\}.$$

In the next theorem, we will prove that $\bar{\mathbf{x}}$ is a global (or unique) minimizer of $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ on the path $\{\mathbf{x}^{(i)}\}_{i=-1}^{u+v}$.

Theorem 3.6. *Let the conditions of Theorem* 3.5 *be satisfied and* $\bar{\mathbf{x}} + \bar{\mathbf{d}} \neq \mathbf{x}^{**}$. *Assume that the path* $\{\mathbf{x}^{(u+j)}\}_{j=1}^{v}$ *satisfies* $\|\mathbf{x}^{(u+j+1)} - \mathbf{x}^*\|_1 > \|\mathbf{x}^{(u+j)} - \mathbf{x}^*\|_1$ *for all* $j \in \{1, \ldots, v-1\}$. *If*

$$0 < \beta < \min\left\{\beta_0, \frac{\beta(j)}{2} : j = 1, \ldots, v-1\right\} \tag{3.14}$$

then $\bar{\mathbf{x}}$ *is the global minimizer of the function H on the path* $\{\mathbf{x}^{(i)}\}_{i=1}^{u+v}$.

Proof. Combining inequality (3.9), it is sufficient to show

$$H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u+j)}; \mathbf{x}^*; \alpha, \beta), \quad j = 1, \ldots, v.$$

Since $\|\mathbf{x}^{(u+j+1)} - \mathbf{x}^*\|_1 > \|\mathbf{x}^{(u+j)} - \mathbf{x}^*\|_1$, $\|\mathbf{x}^{(u+j+1)} - \mathbf{x}^*\|_1 - \|\mathbf{x}^{(u+j)} - \mathbf{x}^*\|_1 = 2$, $j \in \{1, \ldots, v-1\}$ from (3.13). It follows from the definition of the function H and (3.12) that

$$H(\mathbf{x}^{(u+j)}; \mathbf{x}^*; \alpha, \beta) - H(\mathbf{x}^{(u+j+1)}; \mathbf{x}^*; \alpha, \beta) = f(\mathbf{x}^{(u+j+1)}) - f(\mathbf{x}^{(u+j)}) + 2\beta \tag{3.15}$$

holds for all $j \in \{1, 2, \ldots, v-1\}$. When β is chosen to satisfy inequality (3.14), the value of the right-hand side of (3.15) is

$$-\beta(j) + 2\beta < 0, \quad j = 1, \ldots, v-1.$$

Thus

$$H(\mathbf{x}^{(u+j)}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u+j+1)}; \mathbf{x}^*; \alpha, \beta), \quad j = 1, \ldots, v-1.$$

Combining (3.11), it follows that

$$H(\bar{\mathbf{x}}; \mathbf{x}^*; \alpha, \beta) < H(\mathbf{x}^{(u+j)}; \mathbf{x}^*; \alpha, \beta), \quad j = 1, \ldots, v.$$

This completes the proof. □

Theorems 3.5 and 3.6 indicate that the function $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ satisfies the third condition of Definition 7. From the analysis above, we can conclude that when $0 < \beta < \beta_0$ and $0 \leqslant \alpha < \beta/(2qM)$, $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ is a discrete filled function of the function f at point $\mathbf{x}^*$on $\mathbb{S}$.

4. The algorithm

The parameters α and β play important role in filled function algorithm for global optimization. A typical approach to select α, β in the literatures is to first give appropriate initial estimations of α, β, and then adjust them step by step in the process of implementing an algorithm. The shortcoming of the approach is that the algorithm needs to restart after α, β are adjusted, which will increase lots of loads of computation. In the algorithm proposed in this paper, α, β can be given independent of variable $\mathbf{x} \in \mathbb{S}$ and need not to be adjusted in the implementation of the algorithm.

4.1. The estimation of α, β

From Theorems 3.2 and 3.5, it is not difficult to find that when α, β satisfy

$$0 < \beta < \beta_0 = \frac{2qM[f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}{f(\bar{\mathbf{x}}) - f(\mathbf{x}^*) + 4qM},$$
$$0 \leqslant \alpha < \frac{\beta}{2qM}, \tag{4.1}$$

$H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$ is a desired filled function of f at $\mathbf{x}^*$ on $\mathbb{S}$, where $\mathbf{x}^*$, $\bar{\mathbf{x}}$, and $\bar{\mathbf{x}} + \bar{\mathbf{d}}$ satisfy (3.6).

Since $f(\bar{\mathbf{x}}) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}}) \leqslant 4qM$, we have

$$0 < f(\bar{\mathbf{x}}) - f(\mathbf{x}^*) + 4qM \leqslant 8qM - [f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})].$$

It follows that

$$\beta_0 = \frac{2qM[f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}{f(\bar{\mathbf{x}}) - f(\mathbf{x}^*) + 4qM} \geqslant \frac{2qM[f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}{8qM - [f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})]}. \tag{4.2}$$

For a given matrix W, let ε denote the precision of calculation in implementing the algorithm, it is clear that $0 < \varepsilon \leqslant 1$ and, from (3.6),

$$f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}}) \geqslant \varepsilon > 0. \tag{4.3}$$

Let

$$g(t) = \frac{2qMt}{8qM - t}, \quad t \in [0, 4qM],$$

then $g(t)$ is a monotone increasing function with respect to t in the interval $[0, 4qM]$. Thus, from (4.2) and (4.3),

$$\beta_0 \geqslant g(f(\mathbf{x}^*) - f(\bar{\mathbf{x}} + \bar{\mathbf{d}})) \geqslant g(\varepsilon) > g(0) = 0.$$

If the value of β is taken as

$$\beta = \beta(W) = \frac{1}{2} g(\varepsilon) = \frac{qM\varepsilon}{8qM - \varepsilon}, \tag{4.4}$$

then $0 < \beta < \beta_0$ holds for any $\mathbf{x} \in \mathbb{S}$, where $\beta(W)$ means that the value of β only depends on the given matrix W and is independent of variable $\mathbf{x} \in \mathbb{S}$. After the value of β is chosen, any value of α satisfying (4.1) can be selected, for instance,

$$\alpha = \alpha(W) = \frac{\beta(W)}{4qM} = \frac{\varepsilon}{4(8qM - \varepsilon)}. \tag{4.5}$$

It is clear that α satisfies $0 \leqslant \alpha < \beta/(2qM) = \beta(W)/(2qM)$ for any $\mathbf{x} \in \mathbb{S}$.

4.2. The statement of the algorithm

In this subsection, we will state the filled function algorithm and the detail of implementing the algorithm will be given in the following subsections. Once a local minimizer, $\mathbf{x}_k^*$ say, of the function f is obtained, initial points from which the minimization of the filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ is started are randomly generated in the neighborhood $N(\mathbf{x}_k^*)$.

Algorithm (DFFA): The Discrete Filled Function Algorithm

Step 0. **(Initialization)**

(1) Input the matrix W, an initial point $\mathbf{x}_0$, the number $n_1 (\leqslant n)$ and set $k := 0$.

(2) Calculate $\beta = \beta(W)$ and $\alpha = \alpha(W)$.

Step 1. Minimize f from $\mathbf{x}_k$ using a local minimization method and obtain a local minimizer $\mathbf{x}_k^*$ of f on $\mathbb{S}$, set $I = 1$, $\widetilde{N} = \{\mathbf{x}_k^*\}$.

Step 2. Randomly generate an initial point $\mathbf{x}_k^i$ in the set $N(\mathbf{x}_k^*) \backslash \widetilde{N}$.

Step 3. Minimize the filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ starting from the point $\mathbf{x}_k^i$. If a point $\bar{\mathbf{x}}$ is obtained, such that either $\bar{\mathbf{x}}$ satisfies $f(\bar{\mathbf{x}}) < f(\mathbf{x}_k^*)$ or $\bar{\mathbf{x}}$ lies in a basin of f lower than the current basin, then set $\mathbf{x}_{k+1} = \bar{\mathbf{x}}$, $k = k + 1$ and goto Step 1.

Step 4. If $I < n_1$, set $\widetilde{N} = \widetilde{N} \cup \{\mathbf{x}_k^i\}$, $I = I + 1$, goto Step 2.

Step 5. **(Termination)** If $I = n_1$, then return $\mathbf{x}_k^*$ and stop.

Remark 1. The index i of $\mathbf{x}_k^i$ in step 2 means that only the ith element of $\mathbf{x}_k^i$ differs from that of $\mathbf{x}_k^*$. Also, the index i is distinct with the counter I. In fact, by the discussion of the Section 4.3, the index i is a function of the counter I.

Remark 2. The termination condition $I = n_1$ indicates that we do not get a better solution than the current $\mathbf{x}_k^*$ after arbitrary n_1 points out of n points in $N(\mathbf{x}_k^*)$ have been used as initial points to minimize $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$.

4.3. Minimizing f using local search

Given a point $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}} \in \mathbb{S}$, 1-neighborhood search is used to find a point $\mathbf{x}^{i_*} \in N(\mathbf{x})$ satisfying $f(\mathbf{x}^{i_*}) = \min\{f(\mathbf{x}^p) : \mathbf{x}^p \in N(\mathbf{x})\}$. If $\mathbf{x}^{i_*} \neq \mathbf{x}$, then replacing $\mathbf{x}$ with $\mathbf{x}^{i_*}$ and repeating the process until a point $\mathbf{x}^*$ satisfying $f(\mathbf{x}^*) = \min\{f(\mathbf{x}) : \mathbf{x} \in N(\mathbf{x}^*)\}$ is found, which indicates that $\mathbf{x}^*$ is a local minimizer of f.

Let $\mathbf{x}^p = (x_1^p, \ldots, x_p^p, \ldots, x_n^p)^{\mathrm{T}} \in N(\mathbf{x})\backslash\{\mathbf{x}\}$ with $x_i^p = x_i$, for all $i \neq p$ and $x_p^p = -x_p$, i.e., only the pth element of $\mathbf{x}^p$ differs from the pth element of $\mathbf{x}$. Let $\delta(p)$ denote the difference between $f(\mathbf{x}^p)$ and $f(\mathbf{x})$,

$$
\begin{aligned}
\delta(p) &= f(\mathbf{x}^p) - f(\mathbf{x}) = (\mathbf{x}^p)^{\mathrm{T}} W \mathbf{x}^p - (\mathbf{x})^{\mathrm{T}} W \mathbf{x} \\
&= -4 \sum_{i=1}^{n} x_i w_{ip} x_p, \quad p = 1, 2, \ldots, n,
\end{aligned} \tag{4.6}
$$

here we use the fact $w_{ii} = 0, i = 1, \ldots, n$. Calculate

$$
\delta(i_*) = \min\{\delta(p) : p = 1, 2, \ldots, n\}. \tag{4.7}
$$

If $\delta(i_*) \geqslant 0$, then $\mathbf{x}$ is a local minimizer of f. Otherwise $\mathbf{x}^{i_*}$ satisfies $f(\mathbf{x}^{i_*}) < f(\mathbf{x}^p)$, for all $p = 1, 2, \ldots, n$, where $\mathbf{x}^p \in N(\mathbf{x})\backslash\{\mathbf{x}\}$, $p = 1, 2, \ldots, n$.

If $\delta(i_*) < 0$ 1-neighborhood local search can be continued at the point $\mathbf{x}^{i_*}$. Assume that the index i_* satisfies (4.7), that is

$$
\mathbf{x}^{i_*} = (x_1^{i_*}, \ldots, x_{i_*}^{i_*}, \ldots, x_n^{i_*})^{\mathrm{T}} = (x_1, \ldots, -x_{i_*}, \ldots, x_n)^{\mathrm{T}}.
$$

Let $\mathbf{x}^{p,i_*}$ be the point in $N(\mathbf{x}^{i_*})$ from which only pth element differs from the pth element of $\mathbf{x}^{i_*}$ for $p = 1, \ldots, n$, $p \neq i_*$, and we denote $\mathbf{x}^{p,i_*} = (x_1, \ldots, -x_p, \ldots, -x_{i_*}, \ldots, x_n)^{\mathrm{T}}$. Then

$$
\begin{aligned}
\delta(i_*, p) &= f(\mathbf{x}^{p,i_*}) - f(\mathbf{x}^{i_*}) = (\mathbf{x}^{p,i_*})^{\mathrm{T}} W (\mathbf{x}^{p,i_*}) - (\mathbf{x}^{i_*})^{\mathrm{T}} W (\mathbf{x}^{i_*}) \\
&= -4 \sum_{i=1}^{n} x_i^{i_*} w_{ip} x_p^{i_*} \\
&= \delta(p) - 8 x_{i_*}^{i_*} w_{i_* p} x_p^{i_*} \\
&= \delta(p) + 8 x_{i_*} w_{i_* p} x_p \quad (p \neq i_*), \\
\delta(i_*, i_*) &= -\delta(i_*),
\end{aligned} \tag{4.8}
$$

where the $\delta(p)$ is given in (4.6). If $\delta(i_*, p) \geqslant 0$, for all $p = 1, 2, \ldots, n$, then $\mathbf{x}^{i_*}$ is a local minimizer of f. Otherwise, we calculate $\delta(i_*, p_*) = \min_p\{\delta(i_*, p) : p = 1, 2, \ldots, n, p \neq i_*\}$, set $\mathbf{x}^{i_*} = \mathbf{x}^{i_*, p_*}$, $i_* = p_*$ and repeat the process above until a local minimizer of f is found. We summary the 1-neighborhood local search in the following Algorithm (A1).

Algorithm (A1): 1-neighborhood local search
Step 1. Given an initial point $\mathbf{x}^0 \in \mathbb{S}$, calculate $f_0 = f(\mathbf{x}^0)$.
Step 2. Calculate

$$\delta(p) = -4\sum_{i=1}^{n} x_i^0 w_{ip} x_p^0, \quad p = 1, 2, \ldots, n.$$

Step 3. Calculate

$$\delta(i_*) = \min\{\delta(p)\colon p = 1, 2, \ldots, n\}.$$

Step 4. If $\delta(i_*) \geqslant 0$, then set $\mathbf{x}^* = \mathbf{x}^0$, and return $\mathbf{x}^*$ as a local minimizer of f, stop.
Step 5. Set $f_0 = f_0 + \delta(i_*)$, and calculate

$$\delta(p) = \begin{cases} \delta(p) + 8x_{i_*}^0 w_{i_* p} x_p^0, & p \neq i_*, \\ -\delta(i_*), & p = i_*. \end{cases}$$

set $x_{i_*}^0 = -x_{i_*}^0$, goto Step 3.

4.4. Generating initial points to minimize H

For any minimizer $\mathbf{x}_k^*$ of f, a point $\mathbf{x}_k^i \in N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$ is called a *good* point in $N(\mathbf{x}_k^*)$, if a point $\overline{\mathbf{x}}$ in a basin of f lower than the current basin can be obtained by minimizing the filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ starting from $\mathbf{x}_k^i$. Let $\mathbb{S}_*$ denote the set of all minimizers of the function f on $\mathbb{S}$. It is clear that if $\mathbf{x}_k^* = \arg\max\{f(\mathbf{x}^*)\colon \mathbf{x}^* \in \mathbb{S}_*\}$, then there may exist more *good* points in $N(\mathbf{x}_k^*)$. On the contrary, if $\mathbf{x}_k^* = \arg\min\{f(\mathbf{x}^*)\colon \mathbf{x}^* \in \mathbb{S}_*\}$, then there does not exist any *good* point in $N(\mathbf{x}_k^*)$. Thus the number of *good* points in $N(\mathbf{x}_k^*)$ can reflect completely the performance of the minimizer $\mathbf{x}_k^*$, that is, less the number of *good* points exists, the better performance the minimizer $\mathbf{x}_k^*$ has.

However, it is still challenging for efficiently validating whether a point in $N(\mathbf{x}_k^*)$ is a *good* point. A typical and direct approach as presented in the literatures to find an initial point for minimizing H is to test one by one each point in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$. Since there are n elements in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$, it may not be advisable for the approach when n is large. Especially, for max-cut problems, it is unlikely to confirm all points in $N(\mathbf{x}_k^*)$ being *good* points or exclude all points in $N(\mathbf{x}_k^*)$ not being *good* points. The approach proposed in this paper is to randomly generate $n_1(\leqslant n)$ points from $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$ as initial points to minimize H, where n_1 is called *sample size*. The probability of each point in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$ sampled is n_1/n, and n_1 does not need to be chosen too close to n via $\mathbb{S}$ is a connect domain. The numerical results in Section 5 (see Table 5) indicate that after the random generating subroutine is added into the filled function algorithm, the better performance of the proposed algorithm is displayed and the costs of computation and CPU-time are greatly reduced.

Now, we state the random process of generating initial points. For given n, assume that we need to draw out randomly n_1 points from n points in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$. Set $I_1 = [n/n_1]$, where $[a]$ denotes the integer part of the real number a. If n/n_1 is an integer, then we can partition the set $N = \{1, \ldots, n\}$ into n_1 disjoint subsets and each subset has I_1 integers, that is $N_1 = \{1, \ldots, I_1\}$, $N_2 = \{I_1 + 1, \ldots, 2I_1\}, \ldots, N_{n_1} = \{(n_1 - 1)I_1 + 1, \ldots, n_1 I_1\}$. Let c be a random number in (0, 1), $i = (I - 1) \cdot I_1 + [c \cdot I_1] + 1 \in N_I$, $I = 1, 2, \ldots, n_1$, and we take $\mathbf{x}_k^i$ as an initial point to minimize $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$, where the index i means the ith element of $\mathbf{x}_k^i$ differs from the ith element of $\mathbf{x}_k^*$. The random number $c \in (0, 1)$ can be generated from the uniform distribution $\mathcal{U}(0, 1)$.

If n/n_1 is a fraction, then set $n_0 = n - n_1 I_1$. Since $n_0 < n_1$, we can also partition the set $N = \{1, \ldots, n\}$ into n_1 disjoint subsets of N, where each subset of the first $n_1 - n_0$ subsets has I_1 integers and each subset of last n_0 subsets has $I_1 + 1$ integers. Then we also take $\mathbf{x}_k^i$ as an initial point to minimize $H(\mathbf{x}; \mathbf{x}^*; \alpha, \beta)$, where the index i has the same meaning as above and for $I = 1, 2, \ldots, n_1 - n_0$,

$$i = (I - 1) \cdot I_1 + [c \cdot I_1] + 1,$$

or for $I = n_1 - n_0 + 1, \ldots, n_1 - n_0 + j, \ldots, n_1$,

$$\begin{aligned} i &= (n_1 - n_0)I_1 + (I_1 + 1)(I - (n_1 - n_0) - 1) + [(I_1 + 1) \cdot c] + 1 \\ &= (I_1 + 1)(I - 1) - (n_1 - n_0) + [(I_1 + 1) \cdot c] + 1, \end{aligned}$$

$c \in (0, 1)$ is a random number generated from $\mathscr{U}(0, 1)$.

4.5. Minimizing the filled function H

In this subsection, we will describe the method of minimizing H in detail. Let $\mathbf{x}_k^* = (x_{k1}^*, \ldots, x_{km}^*, \ldots, x_{kn}^*)^{\mathrm{T}} \in \mathbb{S}$ denote the current local minimizer of f in problem (MC) and $\overline{\mathbf{x}} = (\overline{x}_1, \ldots, \overline{x}_n)^{\mathrm{T}} \in \partial N(\mathbf{x}_k^*, K)$, $1 \leqslant K \leqslant n$. Denote $\mathscr{X}_K = \{\mathbf{y} \in \mathbb{S}: \mathbf{y} \in N(\overline{\mathbf{x}}), \|\mathbf{y} - \mathbf{x}_k^*\|_1 > \|\overline{\mathbf{x}} - \mathbf{x}_k^*\|_1\}$, then we have the following result.

Lemma 4.1. *There only exist $(n - K)$ elements in set $\mathscr{X}_K$.*

Proof. Without loss of generality, assume that $\overline{\mathbf{x}}$ satisfies

$$\begin{aligned} &\overline{x}_s = -x_{ks}^*, \quad s = 1, 2, \ldots, K, \\ &\overline{x}_s = x_{ks}^*, \quad s = K + 1, K + 2, \ldots, n. \end{aligned}$$

For any $\mathbf{y} \in \mathscr{X}_K$, since $\mathbf{y} \in N(\overline{\mathbf{x}})$, either there exists an index $s \in \{1, \ldots, K\}$ or $s \in \{K+1, \ldots, n\}$, such that $y_s = -\overline{x}_s$. If $s \in \{1, \ldots, K\}$, then there are only $K - 1$ different elements between $\mathbf{y}$ and $\mathbf{x}_k^*$. It follows that $\|\mathbf{y} - \mathbf{x}_k^*\|_1 < \|\overline{\mathbf{x}} - \mathbf{x}_k^*\|_1$ that contradicts $\mathbf{y} \in \mathscr{X}_K$. Hence $s \in \{K + 1, \ldots, n\}$, and each $s \in \{K + 1, \ldots, n\}$ corresponds to an element in $\mathscr{X}_K$. That shows that there are only $n - K$ elements in $\mathscr{X}_K$, i.e., $|\mathscr{X}_K| = n - K$. □

Let $\mathbf{x}_k^i = (x_{k1}^i, \ldots, x_{km}^i, \ldots, x_{kn}^i)^{\mathrm{T}}$ be a point that is randomly generated in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$, where the superscript i expresses that the sign of only the ith element of $\mathbf{x}_k^i$ differs from the sign of the ith element of $\mathbf{x}_k^*$. Since $\mathbf{x}_k^i \in \partial N(\mathbf{x}_k^*, 1)$, $|\mathscr{X}_1| = n - 1$, where $\mathscr{X}_1 = \{\mathbf{x} \in \mathbb{S} : \mathbf{x} \in N(\mathbf{x}_k^i), \|\mathbf{x} - \mathbf{x}_k^*\|_1 > \|\mathbf{x}_k^i - \mathbf{x}_k^*\|_1\}$. Denote $\mathscr{X}_1 = \{\mathbf{x}_k^{i1}, \ldots, \mathbf{x}_k^{i,i-1}, \mathbf{x}_k^{i,i+1}, \ldots, \mathbf{x}_k^{ij}, \ldots, \mathbf{x}_k^{in}\}$, where $\mathbf{x}_k^{ij} = (x_{k1}^{ij}, \ldots, x_{km}^{ij}, \ldots, x_{kn}^{ij})^{\mathrm{T}} \in \mathscr{X}_1$ and $x_{km}^{ij} = x_{km}^i$ $(m = 1, 2, \ldots, n, m \neq j)$, $x_{kj}^{ij} = -x_{kj}^i$.

Now, we present the process of minimizing the filled function H starting from an initial point $\mathbf{x}_k^i \in N(\mathbf{x}_k^*)$. Assume that $f(\mathbf{x}_k^{ij}) \geqslant f(\mathbf{x}_k^*)$ holds for all points $\mathbf{x}_k^{ij} \in \mathscr{X}_1$ $(j = 1, \ldots, n, j \neq i)$ (if there exists $\mathbf{x}_k^{ij} \in \mathscr{X}_1$ such that $f(\mathbf{x}_k^{ij}) < f(\mathbf{x}_k^*)$, then $\mathbf{x}_k^{ij}$ is in a basin lower than the current basin containing $\mathbf{x}_k^*$). Since $f(\mathbf{x}_k^i) \geqslant f(\mathbf{x}_k^*)$, $\|\mathbf{x}_k^{ij} - \mathbf{x}_k^*\|_1 > |\mathbf{x}_k^i - \mathbf{x}_k^*\|_1$, and $\alpha(W)$, $\beta(W)$ are calculated by (4.4), (4.5), respectively, it follows from Lemma 3.3 that

$$\delta_H(j) = H(\mathbf{x}_k^{ij}; \mathbf{x}_k^*; \alpha(W), \beta(W)) - H(\mathbf{x}_k^i; \mathbf{x}_k^*; \alpha(W), \beta(W)) < 0, \quad j = 1, \ldots, n, \quad j \neq i.$$

On the other hand,

$$\begin{aligned} \delta_H(j) &= \alpha(W)[f(\mathbf{x}_k^{ij}) - f(\mathbf{x}_k^i)] - \beta(W)[\|\mathbf{x}_k^{ij} - \mathbf{x}_k^*\|_1 - \|\mathbf{x}_k^i - \mathbf{x}_k^*\|_1] \\ &= -4\alpha(W) \sum_{m=1}^{n} x_{km}^i w_{mj} x_{kj}^i - 2\beta(W) \\ &= \alpha(W)\delta_f(j) - 2\beta(W), \quad j = 1, \ldots, n, \quad j \neq i, \end{aligned}$$

where $\delta_f(j) = f(\mathbf{x}_k^{ij}) - f(\mathbf{x}_k^i)$. Thus we can find an index j to minimize $\delta_H(j)$, that is equivalent to minimize $\delta_f(j)$ for all $j \in \{1, 2, \ldots, n\}\backslash\{i\}$. Let

$$j_* = \arg\min_j\{\delta_f(j): j \in \{1, 2, \ldots, n\}\backslash\{i\}\}$$

then for all $\mathbf{x}_k^{ij} \in \mathscr{X}_1$, we have

$$H(\mathbf{x}_k^{ij_*}; \mathbf{x}_k^*; \alpha(W), \beta(W)) \leqslant H(\mathbf{x}_k^{ij}; \mathbf{x}_k^*; \alpha(W), \beta(W)),$$

and $\mathbf{x}_k^{ij_*}$ is used as the next iterate point for minimizing $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$.

After $\mathbf{x}_k^{ij_*} \in \partial N(\mathbf{x}_k^*, 2)$ is found, we consider the points in set $\mathscr{X}_2 = \{\mathbf{x} \in \mathbb{S}: \mathbf{x} \in N(\mathbf{x}_k^{ij_*}), \|\mathbf{x} - \mathbf{x}_k^*\|_1 > \|\mathbf{x}_k^{ij_*} - \mathbf{x}_k^*\|_1\}$ and $|\mathscr{X}_2| = n - 2$. There are two possible cases.

1. If there exists a point $\mathbf{x} \in \mathscr{X}_2$, such that $f(\mathbf{x}) < f(\mathbf{x}_k^*)$, then $\mathbf{x}$ is a point in a basin of f lower than the current basin and we can re-minimize f using Algorithm (A1) from $\mathbf{x}$.
2. If for all $\mathbf{x} \in \mathscr{X}_2$, $f(\mathbf{x}) \geqslant f(\mathbf{x}_k^*)$, then we can find a point $\bar{\mathbf{x}} \in \mathscr{X}_2$ satisfying $H(\bar{\mathbf{x}}; \mathbf{x}_k^*; \alpha(W), \beta(W)) \leqslant H(\mathbf{x}_k^{ij_*}; \mathbf{x}_k^*; \alpha(W), \beta(W))$. If there is a direction $\bar{\mathbf{d}} \in \{\bar{\mathbf{d}} \in \mathscr{D} : \bar{\mathbf{x}} + \bar{\mathbf{d}} \in \mathbb{S}, \|\bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}_k^*\|_1 > \|\bar{\mathbf{x}} - \mathbf{x}_k^*\|_1\}$, such that $H(\bar{\mathbf{x}} + \bar{\mathbf{d}}; \mathbf{x}_k^*; \alpha(W), \beta(W)) > H(\bar{\mathbf{x}}; \mathbf{x}_k^*; \alpha(W), \beta(W))$, then $\bar{\mathbf{x}}$ is a minimizer of the filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ on the path connecting $\mathbf{x}_k^*$ to $\bar{\mathbf{x}} + \bar{\mathbf{d}}$. Hence $\bar{\mathbf{x}}$ is a point in a basin of f lower than the current basin, and we can also minimize f using Algorithm (A1) from $\bar{\mathbf{x}}$. If for all direction $\bar{\mathbf{d}} \in \{\bar{\mathbf{d}} \in \mathscr{D} : \bar{\mathbf{x}} + \bar{\mathbf{d}} \in \mathbb{S}, \|\bar{\mathbf{x}} + \bar{\mathbf{d}} - \mathbf{x}_k^*\|_1 > \|\bar{\mathbf{x}} - \mathbf{x}_k^*\|_1\}$, $H(\bar{\mathbf{x}} + \bar{\mathbf{d}}; \mathbf{x}_k^*; \alpha(W), \beta(W)) \leqslant H(\bar{\mathbf{x}}; \mathbf{x}_k^*; \alpha(W), \beta(W))$, then we can define the set $\mathscr{X}_3$ and continue to minimize $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ in $\mathscr{X}_3$ from $\bar{\mathbf{x}}$ with the same $(\alpha(W), \beta(W))$.

Formally, the process of minimizing the filled function H can be described as follows.

Algorithm (A2): Minimizing the Filled Function H

Step 1. Let $\mathbf{x}_k^i \in N(\mathbf{x}_k^*)\backslash\widetilde{N}$ be a randomly generated point, set $\widetilde{\mathbf{x}}_k^i = \mathbf{x}_k^i$. Calculate

$$\delta_0 = f(\widetilde{\mathbf{x}}_k^i) - f(\mathbf{x}_k^*) = -4\sum_{m=1}^{n} x_{km}^* w_{mi} x_{ki}^* = -4[(\mathbf{x}_k^*)^{\mathrm{T}}\mathbf{w}_i]x_{ki}^*,$$

set $K = 1$ and $\widetilde{I} = \{i\}$.

Step 2. If $K = [\frac{n}{2}]$, goto step 2 of Algorithm (SA). Otherwise, for all $\mathbf{x}_k^{ij} \in \mathscr{X}_K = \{\mathbf{x} \in N(\widetilde{\mathbf{x}}_k^i): \|\mathbf{x} - \mathbf{x}_k^*\|_1 > \|\widetilde{\mathbf{x}}_k^i - \mathbf{x}_k^*\|_1\}$.
Calculate

$$\delta_f(j) = f(\mathbf{x}_k^{ij}) - f(\widetilde{\mathbf{x}}_k^i) = -4[(\widetilde{\mathbf{x}}_{km}^i)^{\mathrm{T}}\mathbf{w}_j]\widetilde{x}_{kj}^i, \quad j = 1, \ldots, n, \quad j \notin \widetilde{I}, \tag{4.9}$$

$$j_* = \arg\min\{\delta_f(j): j \in \{1, 2, \ldots, n\}\backslash\widetilde{I}\}, \tag{4.10}$$

$$\delta_K(j_*) = f(\mathbf{x}_k^{ij_*}) - f(\mathbf{x}_k^*) = \delta_{K-1}(j_*) + \delta_f(j_*).$$

Here, $\delta_0(j_*) = \delta_0$.

Step 3. If $\delta_K(j_*) = f(\mathbf{x}_k^{ij_*}) - f(\mathbf{x}_k^*) < 0$, then goto Algorithm (A1) to minimize f starting from $\mathbf{x}_k^{ij_*}$.

Step 4. Set

$$\widetilde{\mathbf{x}}_k^i = \mathbf{x}_k^{ij_*}, \quad K = K + 1, \quad \widetilde{I} = \widetilde{I} \cup \{j_*\},$$

goto step 2.

Remark 3. By the symmetry structure of the set $\mathbb{S}$, for any integer K with $[n/2] < K \leqslant n$, if $\mathbf{x} \in \partial N(\mathbf{x}_k^*, K)$, then $-\mathbf{x} \in \partial N(\mathbf{x}_k^*, n - K)$ also $f(-\mathbf{x}) = f(\mathbf{x})$. Thus if we cannot find a point $\bar{\mathbf{x}}$ in a basin of f lower than the current basin along a steepest descent path of $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ from point $\mathbf{x}_k^i \in N(\mathbf{x}_k^*)$ to some point in $\partial N(\mathbf{x}_k^*, [n/2])$, then it is not necessary to continue the search in $\partial N(\mathbf{x}_k^*, [n/2] + 1)$. Hence, when $K = [n/2]$ in step 2, we need to restart the minimization of filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$ by generating another initial point.

Remark 4. In order to obtain the next initial iterate point $\mathbf{x}_k^{ij_*}$ from previous initial point $\widetilde{\mathbf{x}}_k^i$ for minimizing H, we need to calculate equality (4.9) and (4.10), that is, only the need to calculate the inner product of two n-dimensional vectors at most $(n - |\widetilde{I}|)$ times, instead of calculating directly the value $f(\mathbf{x}_k^{ij})$ $(j \in \{1, 2, \ldots, n\}\backslash\widetilde{I})$.

5. Numerical results

In this section, some experimented results are reported on the following two classes of test max-cut problems:

- Randomly generated graphs ($20 \leqslant n \leqslant 200$).
- 12 G-set graphs ($n \geqslant 800$).

When $n_1 = n$, in Algorithm DFFA, it means that in case necessary all points in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$ will be generated as initial points to minimize the filled function $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$. We call the algorithm as Algorithm (DA).

The first class of test problems are generated by the matlab code

$$c = floor(w * abs(full(spRANDSYM(n, a)))),$$
$$W = c - \mathrm{diag}(\mathrm{diag}(c)), \tag{5.1}$$

where the parameter a reveals the density of nonzero entries in matrix W and w reflects the weighted values of matrix W.

When $20 \leqslant n \leqslant 100$, we randomly generate 18 problems with $w = 1$, denoted by S1, ..., S18 and call them as *S-set* problems, and only the Algorithm (DA) is implemented for all *S-set* problems. When $120 \leqslant n \leqslant 200$, we randomly generate eight-problems with w as a random integer in [1,50], denoted by L1, ..., L8 and call them as *L-set* problems. We implement Algorithm(DFFA) with $n_1 = [n/2]$ and Algorithm (DA) for all *L-set* problems. The 12 G-set graphs are G1, G2, G3, G11, G12, G13, G14, G15, G16, G43, G44 and G45 that were created by using a graph generator, *rudy*, written by Pro. Rinaldi. We only implement Algorithm (DFFA) with $n_1 = [n/10]$ for the 12 G-set problems. All the results presented in this section for the proposed algorithm are implemented in a 1.6 GHz Pentium IV personal computer with 256 Mb of RAM.

The elements w_{ij} of matrix W are integers for all test problems. Thus we take $\varepsilon = 1$ in Eqs. (4.4) and (4.5). By (4.4), the value β decreases and tends to $\frac{1}{8}$ with the increasing of the product qM. Hence, a simple choice for the value of β is $\beta = \frac{1}{8}$ that satisfies $\beta < \beta_0$ for all test problems. Thus, we only need to calculate the value of α by (4.5) for different test problems. These values are given in Tables 1–3. It can be seen from Tables 1–3 that with the increasing of the dimension or the density of matrix W, α is decreasing via the value q and M increasing.

Table 1
The values of parameter α for randomly generated the 18 S-set problems

Problem	n	a	$\alpha(W)$
S1	20	0.09	0.0052
S2		0.30	0.0013
S3		0.60	8.9446e−4
S4	30	0.09	0.0070
S5		0.30	1.3055e−3
S6		0.60	8.1300e−4
S7	40	0.09	0.0052
S8		0.30	8.9446e−4
S9		0.60	2.1786e−4
S10	60	0.09	1.7422e−3
S11		0.30	3.3444e−4
S12		0.60	2.0564e−4
S13	80	0.09	1.3155e−3
S14		0.30	2.4814e−4
S15		0.60	1.3024e−4
S16	100	0.09	8.6956e−4
S17		0.30	1.6717e−4
S18		0.60	6.2570e−5

Table 2
The values of parameter α for randomly generated the 8 L-set problems

Problem	n	a	w	$\alpha(W)$
L1	120	0.30	47	3.5568e−7
L2		0.60	7	1.2084e−6
L3	160	0.30	30	3.6837e−7
L4		0.60	14	3.0114e−7
L5	180	0.30	47	1.9473e−7
L6		0.60	9	4.0575e−7
L7	200	0.30	17	4.6358e−7
L8		0.60	46	5.5325e−8

Table 3
The values of parameter α for the 12 G-set problems

Problem	n	Density (%)	$\alpha(W)$
G1	800	6.12	1.3334e−5
G2			1.4348e−5
G3			1.3127e−5
G11	800	0.63	0.0040
G12			0.0040
G13			0.0040
G14	800	1.58	3.5870e−6
G15			2.6710e−6
G16			4.1412e−6
G43	1000	2.10	5.5658e−5
G44			4.5658e−5
G45			4.5658e−5

For each of all *S-set* and *L-set* problems, we run Algorithms (DFFA) (or (DA)) and Continuation Algorithm [30] 10 times, respectively. The numerical results and comparisons are listed in Tables 4 and 5. In both the tables, the columns headed with F^* and F_C^* present the largest values to max-cut problems generated by Algorithms (DFFA) (or (DA)) and Continuation Algorithm in 10 tests, respectively. N and NH denote the number of local minimizers of f which is found by Algorithm (DFFA) or (DA) and the number of mean times minimizing the filled function H, that is, the number of mean initial points generated for minimizing H 10 times, respectively. The CPU-time, denoted by time(s), is the mean value of CPU-time in 10 tests. F_0^* stands for the max-cut value associated with the first minimizer $\mathbf{x}_0^*$ of f obtained by Algorithm (A1) starting from an initial point $\mathbf{x}_0$, where the initial point $\mathbf{x}_0$ is generated randomly by the procedure $\mathbf{x}_0 := \text{sign}(\text{unifrnd}(-1, 1, n, 1))$, where $\text{unifrnd}(-1, 1, n, 1)$ is a Matlab function which generates an n-dimensional uniformly distributed vector whose elements lie in $(-1, 1)$. $\text{sign}(x)$ is a sign function that takes 1 when $x \geqslant 0$ and -1 when $x < 0$.

In order to see how the value F^* is close to the global optimal value of the max-cut problem, we calculate the ratio $S_U\% = (\sum F^* / \sum S_U)\%$, where S_U is an upper bound of F^*, which is generated by solving problem (SDP) using the Matlab software package for solving semidefinite programming, SDPPACK [1].

Table 6 gives the results and comparisons between the hybrid GRASP-VNS [11] (hybrid greedy randomized adaptive search procedure with variable neighborhood search) heuristic, denoted as *gvns*, and the Algorithm (DFFA) on the 12 G-set large size test problems. In the table, the columns headed with gvns present the approximate values to max-cut problems generated by the gvns heuristic and these values are quoted from [11]. The upper bound S_U for the 12 G-set

Table 4
The numerical comparisons of continuous method with Algorithm (DA) for the 18 S-set problems with running continuous method, Algorithm (DA) 10 times, respectively, for each problem

Problem	S_U	Continuous		Algorithm (DA)				
		F_C^*	Time (s)	F_0^*	N	NH	F^*	Time (s)
S1	9	9	0.01	9	1	20	9	0.12
S2	27	27	0.01	25	2	31	27	0.13
S3	43	40	0.02	38	3	26	**43**	0.12
S4	14	12	0.03	11	3	36	**14**	0.30
S5	37	34	0.05	34	3	40	**36**	0.36
S6	74	69	0.06	66	3	36	**72**	0.41
S7	29	26	0.10	25	3	45	**29**	0.92
S8	72	66	0.13	61	4	60	**71**	0.94
S9	124	116	0.15	110	6	56	**120**	0.98
S10	55	46	0.25	49	4	69	**54**	3.57
S11	164	155	0.33	144	8	85	**159**	5.10
S12	311	293	0.46	289	7	88	**300**	3.75
S13	97	90	1.42	89	7	73	**96**	10.32
S14	268	249	1.87	239	7	115	**256**	8.98
S15	465	437	2.02	426	5	106	**452**	8.54
S16	153	142	3.56	139	8	181	**148**	29.70
S17	409	379	4.13	370	7	163	**390**	22.38
S18	762	715	4.67	706	7	178	**736**	23.62
Sum	3113	2905					3012	
S_U%	100	93.32					96.76	

Bold value represents, for the same as test problem, the largest max-cut value obtained by algorithm DFFA, DA, CA or GRASP-VNS.

problems comes from [7]. The value F^* is obtained by running Algorithm (DFFA) only one time. The following observations can be made based on the results in Tables 4–6.

(1) For different density graphs, the proposed filled algorithm is efficient for solving max-cut problems. Comparison with F_C^* and gvns, the value F^* is obviously greatly improved.
(2) Although Algorithm (DFFA) uses less points in $N(\mathbf{x}_k^*)\backslash\{\mathbf{x}_k^*\}$ than Algorithm (DA) as initial points to minimizing $H(\mathbf{x}; \mathbf{x}_k^*; \alpha(W), \beta(W))$, the value F^* and the ratio S_U% obtained by Algorithm (DFFA) are not less than that obtained by Algorithm (DA). Moreover, Algorithm (DFFA) has obtained a better solution than Algorithm (DA) for L4. Especially, Algorithm (DFFA) only needs to spend almost half of CPU-time of Algorithm (DA).
(3) The ratio S_U% reflects that the obtained solution by the proposed algorithm is very close to the global solution for all the test problems.
(4) The value $\rho = (F^* - F_0^*)/N$ listed in the final column of Table 6 indicates that the speed of improving from F_0^* to F^* is fast for problems G1, G2, G3, G43, G44 and G45, which also reflects the filled function algorithm is promising for solving max-cut problems.
(5) Although we have greatly reduced the computation cost by random generating initial points and avoiding to calculate the function value repeatedly, the CPU-time is still large. This is because N, the number of the local minimizers of the function f and the times of minimizing the filled function H are large. That is, the larger the N and NH are, the more the CPU-time is.

6. Conclusions

A discrete filled function algorithm is proposed to find approximate global solutions for NP-hard max-cut problems. The algorithm is implemented via two phase cycle: in the first phase, the objective function f in problem (MC) is

Table 5
The numerical comparisons of continuous method with Algorithm (DA), (DFFA) for the 8 L-set problems with running continuous method, Algorithm (DA) and (DFFA) 10 times, respectively, for each problem

Problem	S_U	Continuous		Algorithm (DA)/(DFFA)				
		F_C^*	Time (s)	F_0^*	N	NH	F^*	Time (s)
L1	49 738	46 423	6.25	46 382	10	194	**47 934**	48.27
				45 128	7	97	**47 934**	23.15
L2	11 765	11 122	6.86	11 066	10	169	**11 421**	40.30
				11 090	13	97	**11 421**	22.15
L3	55 241	51 525	7.67	50 920	9	194	**53 311**	81.50
				51 451	10	127	**53 311**	52.10
L4	43 860	41 690	8.77	41 610	7	220	42 459	92.21
				41 654	9	144	**42 477**	58.90
L5	108 190	102 082	9.80	100 460	11	292	**104 493**	159.02
				102 033	9	149	**104 493**	79.05
L6	33 856	32 078	9.72	31 975	16	285	**33 073**	151.74
				31 812	13	128	**33 073**	66.87
L7	46 810	44 122	11.56	43 814	20	321	**45 182**	214.87
				44 072	18	185	**45 182**	121.29
L8	225 610	215 988	14.02	215 972	14	351	**219 461**	237.69
				214 798	11	222	**219 461**	148.97
Sum	575 070	545 030					557 334	
							557 352	
$\%S_U$	100	94.77					96.91	
							96.92	

Bold value represents, for the same as test problem, the largest max-cut value obtained by algorithm DFFA, DA, CA or GRASP-VNS.

Table 6
The numerical comparisons of gvns heuristic with Algorithm (DFFA) for 12 G-set problems with only running Algorithm (DFFA) once time for each problem

Problem	S_U	gvns	Algorithm (DFFA)					
			F_0^*	N	NH	F^*	Time (s)	ρ
G1	12 078	11 475	11 355	14	165	**11 557**	1112.32	14.43
G2	12 084	11 499	11 280	21	196	**11 563**	1210.40	13.46
G3	12 077	11 507	11 315	19	217	**11 549**	1254.53	12.32
G11	627	544	448	40	218	**557**	1316.25	2.73
G12	621	542	418	41	227	**550**	1478.90	3.23
G13	645	572	430	37	202	**576**	1099.36	3.95
G14	3187	3009	2926	21	198	**3029**	1016.42	4.90
G15	3169	3008	2891	28	214	**3021**	1103.71	4.64
G16	3172	2983	2896	22	169	**3025**	922.49	5.86
G43	7027	6583	6224	27	285	**6587**	1452.32	13.44
G44	7022	6559	6373	22	293	**6578**	1546.18	9.32
G45	7020	6553	6372	25	267	**6573**	1402.56	8.04
Sum	68 729	64 834				65 165		
$\%S_U$	100	94.33				94.81		

Bold value represents, for the same as test problem, the largest max-cut value obtained by algorithm DFFA, DA, CA or GRASP-VNS.

minimized using the 1-neighborhood local search to obtain a local minimizer, and then in the second phase, the discrete filled function from some neighbor points of the local optimizer is minimized and the two cycles are repeated until the stop conditions are satisfied. The properties of the proposed filled function are analyzed. The characteristics of max-cut problems are used to show that the parameter values in the filled function need not be adjusted. This greatly increases the efficiency of the proposed filled function method. Numerical results and comparisons are reported to indicate that the filled function algorithm is efficient for max-cut problems. For large scale graphs, the algorithm may spend more time to find a desired solution which is due to the number of the local minimizers of the function f and the times of minimization of the filled function H from different initial points in a neighbor of a local minimizer of f. Hence further works on the proposed algorithm are required to refine the algorithm in theory and implementation.

Acknowledgments

The authors are indebted to two anonymous referees whose comments helped considerably to improve this paper.

References

[1] F. Alizadeh, J. Haeberly, M. Nayakkankuppam, M. Overton, S. Schmieta, SDPPACK use's guide: version 0.9 beta for MATLAB5.0, 1997.
[2] H. Alperin, I. Nowak, Lagrangian smoothing heuristics for max-cut, J. Heuristics 11 (2005) 447–463.
[3] F. Barahona, Grötschel, G. Reinelt, An application of combinatorial optimization to statistical physics and circuit layout design, Oper. Res. 36 (1988) 493–513.
[6] D. Bertsimas, Y. Ye, Semidefinite relaxations, multivariate normal distributions, and order statistics, in: D.-Z. Du, P. Pardalos (Eds.), Handbook of Combinatorial Optimization, Kluwer Academic Publishers, Dordrecht, 1998, pp. 1–17.
[7] S. Burer, R.D.C. Monteiro, A projected gradient algorithm for solving the maxcut SDP relaxation, Optimization Methods and Software 15 (2001) 175–200.
[9] S. Burer, R.D.C. Monteiro, Y. Zhang, Rank-two relaxation heuristics for max-cut and other binary quadratic programs, SIAM J. Optim. 12 (2001) 503–521.
[11] P. Festa, P.M. Pardalos, M.G.C. Resende, C.C. Ribeiro, Randomized heuristics for the max-cut problem, Optimization Methods and Software 17 (2002) 1033–1058.
[12] M. Garey, D. Johnson, L. Stochmeter, Some simplified NP-complete graph problems, Theoret. Comput. Sci. 1 (1976) 237–267.
[13] R.-P. Ge, A filled function method for finding a global minimizer of a function of several variables, Math. Programming 46 (1990) 191–204.
[14] R.-P. Ge, C.B. Huang, A continuous approach to nonlinear integer programming, Appl. Math. Comput. 34 (1) (1989) 39–60.
[16] M.X. Goemans, D.P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, J. Assoc. Comput. Mach. 42 (6) (1995) 1115–1145.
[17] Y.H. Gu, Z.Y. Wu, A new filled function method for nonlinear integer programming problem, Appl. Math. Comput. 173 (2006) 938–950.
[18] X.-L. He, C.-X. Xu, C.-C. Zhu, A new class of filled functions for global minimization, computation intelligence and security, part I, in: Proceedings Lecture Notes in Artificial Intelligence, vol. 3801, 2005, pp. 1088–1093.
[20] R.M. Karp, Reducibility among combinatorial problems, in: R. Miller, J. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85–103.
[21] X. Liu, Finding global minima with a computable filled function, J. Global Optim. 19 (2001) 151–161.
[23] C.-K. Ng, L.-S. Zhang, D. Li, W.-W. Tian, Discrete filled function method for discrete global optimization, Comput. Optim. Appl. 31 (2005) 87–115.
[26] Y.-L. Shang, L.-S. Zhang, A filled function method for finding a global minimizer on global integer optimization, J. Comput. Appl. Math. 181 (2005) 200–210.
[27] C.-X. Xu, X.-L. He, F.-M. Xu, An effective continuous algorithm for approximate solutions of large scale max-cut problems, J. Comput. Math. 24 (6) (2006) 749–760.
[30] F.-M. Xu, C.-X. Xu, H.-G. Xue, A feasible direction algorithm without line search for solving max-bisection problem, J. Comput. Math. 23 (2005) 619–634.
[31] Z. Xu, H.-X. Huang, P.M. Pardalos, C.X. Xu, Filled functions for unconstrained global optimization, J. Global Optim. 20 (2001) 49–65.
[32] L.-S. Zhang, C. Ng, D. Li, W.-W. Tian, A new filled function method for global optimization, J. Global Optim. 28 (2004) 17–43.
[33] W.-X. Zhu, An approximate algorithm for nonlinear integer programming, Appl. Math. Comput. 93 (2–3) (1998) 183–193.
[34] U. Zwick, Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to max cut and other problems, in: Proceedings of 31st STOC, 1999, pp. 679–687.