A quantum method for dynamic nonlinear programming technique using Schrödinger equation and Monte Carlo approach

Amandeep Kaur*,‡, Satnam Kaur†,§ and Gaurav Dhiman*,¶

*Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala 147004, Punjab, India
†Department of Computer Science and Engineering, Dr BR Ambedkar National Institute of Technology, Jalandhar 144011, Punjab, India
‡kaur.amandeep@thapar.edu
§satnamk.cs.14@nitj.ac.in
¶gaurav.dhiman@thapar.edu

Received 5 July 2018
Revised 9 September 2018
Accepted 14 September 2018
Published 26 October 2018

The power of quantum computing may allow for solving the problems which are not practically feasible on classical computers and suggest a considerable speed up to the best known classical approaches. In this paper, we present the contemporary quantum behaved approach which is based on Schrödinger equation and Monte Carlo method. The three basic steps of proposed technique are also mathematically modeled and discussed for effective movement of particles. The performance of the proposed approach is tested for solving the dynamic nonlinear problem. Experimental results reveal the supremacy of proposed approach for solving the nonlinear problem as compared to other approaches.

Keywords: Quantum; nonlinear; Schrödinger; Monte Carlo; particles.

1. Introduction

Quantum computers are used to improve the speed of quantum algorithms using some quantum mechanics laws. Nowadays, entanglement is a theoretical concept to provide the speed comparison between classical counterparts. In recent times, many scientists have studied entanglement and its related disciplines measures. The main application of quantum computing is in solving optimization problems. Other application areas of this problem can be listed as dynamic nonlinear problem using low computational efforts. However, optimization is the process of determining the decision variables of a function to minimize or maximize its values. Most of the real-world problems1-7 include nonlinear constraints, non-convex, complicated and
large number of solution spaces. Therefore, solving such problems with large number of variables and constraints is very tedious and complex. There are many local optimum solutions which do not guarantee the best overall solution using classical numerical methods.

To overcome such problems, metaheuristic optimization algorithms have been introduced which are capable of solving such complex problems during the course of iterations. Recently, immense interest has been focused on the development of metaheuristic algorithms owing to their flexibility and simplicity by nature.

Metaheuristics are broadly classified into two categories such as single solution and population-based algorithms. Single solution-based algorithms are those in which a solution is randomly generated and improved until the optimum result is obtained. Whereas, population-based algorithms are those in which a set of solutions is randomly generated in a given search space and solution values are updated during iterations until the best solution is generated.

However, single solution-based algorithms may trap into local optima which may prevent us to find global optimum as it reforms only one solution which is randomly generated for a given problem. On the other hand, population-based algorithms have inherent ability to escape local optima. Due to this, nowadays, population-based algorithms have gained the attention of multitudinous researchers.

The categorization of population-based algorithms is done on the basis of theory of evolutionary algorithms, physics law-based algorithms, swarm intelligence of particles, and biological behavior of bio-inspired algorithms. Evolutionary algorithms are inspired by the evolutionary processes such as reproduction, mutation, recombination and selection. These algorithms are based on the survival fitness of the candidate in a population (i.e. a set of solutions) for a given environment. The physics law-based algorithms are inspired by physical processes according to some physics rules such as gravitational force, electromagnetic force, inertia force, heating and cooling of materials. Swarm intelligence-based algorithms are inspired by the collective intelligence of swarms.

A well-known algorithm of swarm intelligence technique is Particle Swarm Optimization (PSO). PSO is inspired by the social behavior of fish schooling or bird flocking. Each particle can move around the search space and update its current position with respect to the global best solution. The reason behind the popularity of this algorithm is that only few parameters are required for fine-tuning.

Every quantum-based optimization algorithm needs to address the exploration and exploitation of a search space and maintain a good balance between exploration and exploitation. The exploration phase investigates the different promising regions in a search space, whereas exploitation searches the near global optimal solutions around the promising regions. Therefore, to acquire the near optimal solutions, fine-tuning of these two phases is required. Despite the significant number of recently developed optimization algorithms, the question that arises is why do we need to develop more optimization techniques. The answer lies in No Free Lunch (NFL) theorem. According to this theorem, the performance of one
optimization algorithm for a specific set of problems does not guarantee to solve other optimization problems because of their different nature. The NFL theorem allows researchers to propose some novel optimization algorithms for solving the problems in various fields.\textsuperscript{25–29} Hence, this paper presents a novel quantum-behaved approach for dynamic nonlinear problem.

The rest of this paper is structured as follows. Section 2 presents the fundamental concepts of recently developed optimization algorithm. Section 3 presents the quantum-based approach in detail. Section 4 describes the constrained handling approach. The real-life constrained industrial optimization problem and its comparison is presented in Sec. 5. Finally, the conclusion is discussed in Sec. 6.

2. Spotted Hyena Optimizer (SHO)

Spotted Hyena Optimizer is a metaheuristic bio-inspired optimization algorithm developed by Dhiman \textit{et al.}\textsuperscript{30–32} The fundamental concept of this algorithm is to simulate the social behaviors of spotted hyenas. There are four main steps of SHO algorithm which are inspired by encircling, hunting, attacking and searching behaviors of spotted hyenas.

2.1. Encircling behavior

The target behavior or objective is considered as the best solution and the other search agents can update their positions with respect to obtained best solution. The mathematical model of this behavior is represented by Eqs. (1) and (2).

\[
D_h = |B \times P_p(x) - P(x)|, \quad (1)
\]
\[
P(x + 1) = P_p(x) - E \times D_h, \quad (2)
\]

where \(D_h\) represents the distance between the behavior and spotted hyena, \(x\) indicates the current iteration, \(B\) and \(E\) are coefficient vectors, \(P_p\) indicates the position vector of behavior, \(P\) is the position vector of spotted hyena.

However, vectors \(B\) and \(E\) are calculated by Eqs. (3)–(5), respectively.

\[
B = 2 \times rd_1, \quad (3)
\]
\[
E = 2h \times rd_2 - h, \quad (4)
\]
\[
h = 5 - \left(\text{Iteration} \times \frac{5}{\text{MaxIteration}}\right), \quad (5)
\]

where \(\text{Iteration} = 0, 1, 2, \ldots, \text{MaxIteration}\).

The \(h\) is linearly decreased from 5 to 0 and \(rd_1, rd_2\) are random vectors in range [0, 1].
2.2. Hunting

The hunting strategy of SHO algorithm is described by Eqs. (6)–(8).

\[ D_h = |B \times P_h - P_k| , \]
\[ P_k = P_h - E \times D_h , \]
\[ C_h = P_k + P_{k+1} + \cdots + P_{k+N} , \]

where \( P_k \) represents the position of other spotted hyenas. However, variable \( N \) indicates the total number of spotted hyenas which are calculated by Eq. (9).

\[ N = \text{count}_{\text{nos}}(P_h, P_{h+1}, P_{h+2}, \ldots, (P_h + M)) , \]

where \( P_h \) defines the position of first best obtained spotted hyena, \( M \) is a random vector in range \([0,1]\), \( \text{nos} \) represents the number of solutions and count all candidate solutions, and \( C_h \) is a group of \( N \) number of optimal solutions.

2.3. Attacking behavior

The mathematical formulation for attacking the behavior is defined by Eq. (10).

\[ P(x + 1) = \frac{C_h}{N} , \]

where \( P(x + 1) \) saves the best solution and updates the positions of other search agents with respect to the position of best search agent.

2.4. Search for behavior

For searching the suitable solution, \( E \) is responsible which is greater than 1 or less than 1 using Eq. (4). \( B \) is another important constituent of SHO algorithm for exploration purpose. It contains random values which provide the random weights of behavior as shown in Eq. (3). To show the more random behavior of SHO algorithm, assume vector \( B > 1 \) preceding \( B < 1 \) to demonstrate the effect in the distance.

The SHO algorithm solves various high-dimensional problems with low computational efforts and avoids local optimum problem. The pseudo code of SHO algorithm is described in Algorithm.

3. Quantum-Based Spotted Hyena Optimizer

In quantum computing, the entanglement of quantum particles is necessary for quantum mechanics. This unique property of quantum has no analog in classical mechanics. Therefore, two particles can be considered as entangled when the energy between them has been large at any time in the past.\(^{33}\) However, none of these particles is involved in the emission of other particles. Entanglement is commonly used in quantum communication, quantum cryptography and quantum computing.\(^{33,34}\) Schrödinger introduced the concept of “entanglement” in quantum mechanics.\(^{35}\) In
Quantum method using Schrödinger equation and Monte Carlo approach

In this work, for simplicity, we consider a search agent in two-dimensional search space and solve the $P(x+1) = C_h/N$ using nonlinear Schrödinger equation as

$$q_i = \phi \times Q_{ij} + (1 - \phi) \times Q_{hi}, \quad 0 < \phi < 1,$$

where $L$ represents the search scope of each particle. Now, obtain the position of particles using the Monte Carlo method as

$$y = q \pm \frac{L}{2} \ln(1/v), \quad v \sim U(0,1),$$

where $v$ is a random number lying in range $[0,1]$. Now, compute the distance between particles as

$$D(x) = \int_{-\infty}^{x} P(x)dx = e^{-2/L},$$

where $\alpha$ is the parameter of the algorithm. Therefore, the quantum mechanism for updating the positions of search agents is defined as

$$P(x+1) = C_h \cdot \frac{y}{N}.$$

The steps and flowchart (see Fig. 1) of the proposed approach is described as follows:

**Step 1:** Initialize the search agents $P_s$, where $s = 1, 2, \ldots, n$.

**Step 2:** Choose the initial parameters: $h, B, E, \alpha$ and $N$ and define the maximum number of iterations.

**Step 3:** Calculate the fitness value of each search agent.

**Step 4:** The best search agent is explored in the given search space.

**Step 5:** Define the group of optimal solutions, i.e. cluster until the satisfactory result is found.

**Step 6:** Update the positions of search agents using Eq. (17).

**Step 7:** Check whether any search agent goes beyond the boundary in a given search space and adjust it.

**Step 8:** Calculate the updated search agent fitness value and update the vector $P_h$, if there is a better solution than previous optimal solution.

**Step 9:** Update the group of spotted hyenas $C_h$ to updated search agent fitness value.

**Step 10:** If the stopping criterion is satisfied, the algorithm will be stopped. Otherwise, return to Step 5.

**Step 11:** Return the best optimal solution, after stopping criteria are satisfied, which is obtained so far.
Fig. 1. Flowchart of proposed quantum approach.
Algorithm: Quantum Spotted Hyena Optimizer

Input: The spotted hyenas population $P_s$ ($s \leftarrow 1, 2, \ldots, n$)

Output: The obtained best search agent

1: procedure QSHO
2: Initialize the parameters $h, B, E$ and $N$
3: Calculate the fitness value of each search agent
4: $P_h \leftarrow$ The best search agent
5: $C_h \leftarrow$ The group or cluster of all far optimal solutions
6: while $(x < \text{MaxIteration})$ do
7:     for each search agent do
8:         Update the position of current agent using Eq. (17)
9:     end for
10:     Update $h, B, E$ and $N$
11:     Check if any search agent goes beyond the given search space and then adjust it
12:     Compute the fitness of each search agent
13:     Update $P_h$ if there is a better solution than previous optimal solution
14:     Update the group $C_h$ with respect to $P_h$
15:     $x \leftarrow x + 1$
16: end while
17: return $P_h$
18: end procedure

4. Constraint Handling

Constraint handling is one of the biggest challenges in solving optimization problems using metaheuristic techniques. There are five constraint handling techniques\textsuperscript{36}: penalty functions, hybrid methods, separation of objective functions and constraints, repair algorithms and special operators. Among these techniques, the penalty functions are simple and easy to implement. There are numerous penalty functions such as static, annealing, adaptive, co-evolutionary and death penalty. These approaches convert constraint problems into unconstraint problem by adding some penalty values. In this paper, a static penalty approach is employed to handle constraints in optimization problems.

$$\zeta(z) = f(z) \pm \left[ \sum_{i=1}^{m} l_i \cdot \max(0, t_i(z))^\alpha + \sum_{j=1}^{n} o_j \cdot |U_j(z)| \right], \quad (18)$$

where $\zeta(z)$ is the modified objective function and $l_i, o_j$ are positive penalty values. This approach assigns the penalty value for each infeasible solution. In the death penalty approach, a large value is assigned to objective function of infeasible solution. Therefore, the static penalty function is employed which helps the search agents to move towards the feasible search space of the problem.
5. Optical Buffer Design

The optical buffer permits the optical CPUs to measure different optical packets by slowing down the group velocity of light. This whole process is executed using the most popular device known as Photonic Crystal Waveguide (PCW). Generally, PCWs have a lattice-shaped structure with a line defect and holes with different radii which yield the characteristics of slow light. In this section, the structure of PCW called a Bragg Slot Photonic Crystal Wave guide (BSPCW) is optimized to achieve these characteristics by QSHO algorithm.

The performance of slow light devices is compared using Delay Bandwidth Product (DBP) and Normalized DBP (NDBP) metrics, which are formulated as\[37:\]

\[
DBP = \Delta d \cdot \Delta b, \tag{19}\]

where \(\Delta d\) and \(\Delta b\) indicate the delay and bandwidth of slow light device, respectively.

\[
NDBP = \overline{m_g} \cdot \Delta \omega / \omega_0, \tag{20}\]

where \(\overline{m_g}\) is the average of group index, \(\Delta \omega\) is the bandwidth, and \(\omega_0\) is the central frequency of light wave. However, NDBP has a relation with group index \((m_g)\) as

\[
m_g = \frac{V}{v_g} = C \frac{dv}{d\omega}, \tag{21}\]

where \(\omega\) is the dispersion, \(v\) defines the wave vector, \(C\) indicates the velocity of light, and \(m_g\) is responsible for changing in the bandwidth range. The average of \(m_g\) is calculated as

\[
\overline{m_g} = \int_{\omega_L}^{\omega_H} m_g(\omega) \frac{d\omega}{\Delta \omega}, \tag{22}\]

since \(m_g\) has a constant value with maximum fluctuation of \(\pm 10\%\).\[38:\] The detailed information about PCWs can be found in Ref. 39. The mathematical formulation of this problem is described as follows:

\[
z = [z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8] = \left[ \frac{R_1}{a}, \frac{R_2}{a}, \frac{R_3}{a}, \frac{R_4}{a}, \frac{R_5}{a}, \frac{l}{a}, w_h, w_l \right],
\]

Maximize: \(f(z) = NDBP\),

subject to:

\[
\max(|\beta_2(\omega)|) < 10^6 a/2\pi c^2,
\]

\[
\omega_H < \min(\omega_{\text{upband}}),
\]

\[
\omega_L > \max(\omega_{\text{lowband}}),
\]

\[
k_n > k_{nH} = \omega_{\text{guidedmode}} > \omega_H,
\]

\[
k_n < k_{nL} = \omega_{\text{guidedmode}} < \omega_L,
\]

where
Quantum method using Schrödinger equation and Monte Carlo approach

\[ \omega_H = \omega(k_n H) = \omega(1.1m_\gamma 0), \quad v\omega_L = \omega(k_n L) = \omega(0.9m_\gamma 0), \]

\[ vk_n = \frac{ka}{2\pi}, \quad \Delta \omega = \omega_H - \omega_L, \quad a = \omega_0 \times 1550 \text{ nm}, \]

\[ 0 \leq z_{1-5} \leq 0.5, \quad 0 \leq z_6 \leq 1, \quad 0 \leq z_{7,8} \leq 1. \]

There are five constraints defined in this problem for QSHO algorithm. The algorithm is iterated for 30 times and the obtained results are tabulated in Table 1. The results reveal the substantial improvements of 99% and 10% in bandwidth (see Fig. 3) using QSHO approach in comparison to the results reported by GWO.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>QSHO</th>
<th>GWO \textsuperscript{40}</th>
<th>Wu \textit{et al.}\textsuperscript{41}</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>0.31345a</td>
<td>0.33235a</td>
<td>—</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>0.22371a</td>
<td>0.24952a</td>
<td>—</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>0.23401a</td>
<td>0.26837a</td>
<td>—</td>
</tr>
<tr>
<td>( R_4 )</td>
<td>0.27676a</td>
<td>0.29498a</td>
<td>—</td>
</tr>
<tr>
<td>( R_5 )</td>
<td>0.30212a</td>
<td>0.34992a</td>
<td>—</td>
</tr>
<tr>
<td>( l )</td>
<td>0.7262a</td>
<td>0.7437a</td>
<td>—</td>
</tr>
<tr>
<td>( w_h )</td>
<td>0.2050a</td>
<td>0.2014a</td>
<td>—</td>
</tr>
<tr>
<td>( w_l )</td>
<td>0.60023a</td>
<td>0.60073a</td>
<td>—</td>
</tr>
<tr>
<td>( a )  (nm)</td>
<td>325</td>
<td>343</td>
<td>430</td>
</tr>
<tr>
<td>( \frac{a}{2\pi c^2} )</td>
<td>( 10^3 )</td>
<td>( 10^3 )</td>
<td>( 10^3 )</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>37.0</td>
<td>33.9</td>
<td>17.6</td>
</tr>
<tr>
<td>NDBP</td>
<td>0.51</td>
<td>0.43</td>
<td>0.26</td>
</tr>
</tbody>
</table>

Fig. 2. The obtained optimized super cell.
Wu et al.\textsuperscript{41} A similar behavior has been observed in the NDBP. The improvements achieved in NDBP (see Fig. 4) are 90\% and 14\% as compared with Wu et al.\textsuperscript{41} and GWO\textsuperscript{40} approaches, respectively. The optimized super cell is shown in Fig. 2. It shows that the optimized structure has a very good bandwidth without band mixing. The results demonstrate that QSHO algorithm proved its merit for solving optical buffer optimization problem.

6. Conclusions
This paper discussed the contemporary quantum-behaved approach that is based on the nonlinear Schrödinger equation and Monte Carlo method. The main concept of the proposed approach is to analyze the behaviors of particles (i.e. search agents) in nature. Furthermore, the proposed quantum technique is employed to solve the dynamic nonlinear problem named as optical buffer design problem. The obtained results are compared with other various competitor algorithms. The results reveal that the proposed technique is an efficient approach to solve these problems and generate the near optimal search agents.
Quantum method using Schrödinger equation and Monte Carlo approach

References