MAKHA – A new hybrid stochastic optimization method for phase equilibrium calculations

Seif-Eddine K. Fateen*, Ahmed M. E. Khalil
Department of Chemical Engineering, Cairo University
Giza, Egypt
* Corresponding author: sfateen@alum.mit.edu

Ragab K. Abdel-Magied
Production Technology Department, Faculty of Industrial Education, Beni-Suef University, Beni-Suef, Egypt

Abstract—The search for reliable and efficient global optimization algorithms for solving phase stability and phase equilibrium problems in applied thermodynamics is an ongoing area of research. In this study, we introduce a new algorithm, MAKHA, which is a hybrid between Monkey Algorithm (MA) and Krill Herd Algorithm (KHA). Its performance is compared with the two original algorithms along with Cuckoo Search for solving difficult phase stability and phase equilibrium problems. The results MAKHA is more reliable than the original two algorithms. However, its reliability could not exceed the reliability of CS. In summary, MAKHA is a promising nature-inspired optimization method to perform applied thermodynamic calculations for process design.

Keywords—Phase Equilibrium; Phase Stability; Stochastic Global Optimization; Monkey Algorithm; Krill Herd Algorithm; Cuckoo Search

I. INTRODUCTION

The sophisticated decision making process that swarms of living organisms exhibit has inspired several optimization algorithms. Examples of these swarm intelligence optimization techniques are based on the decision making process of fireflies, ants, bees or birds [1]. In general, the bio-inspired methods are quite simple to implement and use. They do not require any assumptions or transformation of the original optimization problems, do not require good starting points, can easily move out of local minima in their path to the global minimum, and can be applied with any model (i.e., black box model), yet provide a high probabilistic convergence to the global optimum. They can often locate the global optimum in modest computational time compared to deterministic optimization methods [2]. Therefore, these techniques are more advantageous compared to traditional local gradient-based and global deterministic optimization techniques.

Recently, swarm intelligence optimization methods have been introduced for solving challenging global optimization problems involved in the thermodynamic modeling of phase equilibrium for chemical engineering applications [3-10]. In particular, the calculations of phase and chemical equilibrium are an essential component of all process simulators in chemical engineering. The prediction of phase behavior of a mixture involves the solution of the two main thermodynamic problems: phase stability (PS) and phase equilibrium calculations (PEC) [11]. PS problems involve the determination of whether a system will remain in one phase at the given conditions or split into two or more phases. This type of problems usually precedes the PEC problem, which involves the determination of the number, type and composition of the phases at equilibrium at the given operating conditions [9]. Note that a reactive phase equilibrium calculation (RPEC) or chemical equilibrium calculation is performed if any reaction is possible in the system under study. During the analysis of a chemical engineering process, PS, PEC and/or RPEC problems usually need to be solved numerous times. Solving these types of thermodynamic problems involves the use of global optimization methods. In particular, PS analysis requires the minimization of the tangent plane distance function (TPDF), while the Gibbs free energy function needs to be minimized for PEC and RPEC subject to the corresponding constraints [9]. For these thermodynamic problems, finding a local minimum is not sufficient; and the global minimum must be identified for determining the correct thermodynamic condition.

In general, the high non-linearity of thermodynamic models, the non-convexity of the objective functions, and the presence of a trivial solution in the search space make PEC, RPEC and PS problems difficult to solve. Moreover, these thermodynamic problems may have local optimal values that are very comparable to the global optimum value, which makes it challenging to find the global optimum [9]. Hence, PS, PEC, and RPEC problems require a reliable and efficient global optimization algorithm. To date, there are no effective optimization methods at all for performing these thermodynamic calculations. Current methods for phase equilibrium modeling have their own deficiencies and sometime fail to find the correct solutions for difficult problems such as the calculation of simultaneous phase and chemical equilibrium for systems containing many components near the critical point of the mixture and the phase boundaries [11]. Wrong estimation of the thermodynamic state may have negative impacts on the design, analysis and operation of such novel processes. Therefore, the search for better methods and techniques to solve these often-difficult thermodynamic problems is still ongoing and new optimization algorithms should be developed and/or analyzed.

Swarm intelligence optimization methods have been insufficiently studied in chemical engineering applications including the thermodynamic modeling of phase equilibrium [3-10]. In particular, this work introduces a new hybrid bio-inspired optimization algorithm for PEC, RPEC and PS problems involving multiple components, multiple phases and popular thermodynamic models. This algorithm is called
MAKHA, which is a hybrid between Monkey Algorithm (MA) [12] and Krill Herd Algorithm (KHA) [13]. After introducing the new algorithm, the study evaluates its performance against the performance of the original two algorithms. In addition, a comparison is made with Cuckoo Search (CS) [14], which has been shown to be the most reliable algorithm for solving phase stability and phase equilibrium problems [9]. We systematically used those methods on some of the difficult phase stability and phase equilibrium problems reported in the literature, and then analyzed their performance in terms of clear reliability and efficiency metrics.

The remainder of this manuscript is organized as follows. The new optimization method is introduced in Section 2. A brief description of the phase stability and equilibrium problems is given in Section 3, including the implementation details of the algorithms. Section 4 presents the results and discussion of their performance in solving these thermodynamic calculations. Finally, the conclusions of this study are summarized in Section 5.

II. DESCRIPTION OF MONKEY ALGORITHM – KRILL HERD ALGORITHM HYBRID

MAKHA is a new hybrid algorithm, which combines some of the mechanisms and processes of MA and KHA together to get a reliable algorithm with appreciated performance. The steps of both algorithms include exploration/diversification and exploitation/intensification features as follows. The exploration/diversification features of MA are the somersault process and the watch-jump process, while for KHA, they are the physical random diffusion and the genetic operators. On the other hand, the exploitation/intensification features of MA are the climb and the watch-jump process, while for KHA, they are the induced motion and the foraging activity.

Both algorithms attempt to balance between exploration/diversification and exploitation/intensification features. MA has two exploration operators and two exploitation operators. The watch-jump process acts as both an exploration and an exploitation operator. The somersault operator is a high-performing diversification operator that makes a good use of the pivot function. Since MA is an exploration-dominant algorithm, the exploitation balance is brought to the algorithm by running the climb process twice per iteration. Increasing the number of cycles needed in MA reduces the computational efficiency as it increases the number of function evaluations (NFE):

Even though KHA also has two exploration operators and two exploitation operators, its exploration component is not dominating because the physical random diffusion is a less efficient exploration operator than the somersault operator is. Thus, entrapment in local minima is more probable in KHA than in MA. The trapping problem can be addressed in the KHA by the use of two genetic operators (cross over and mutation). Since the foraging movement is a high-performing exploitation operator, KHA could be considered an exploitation-dominant algorithm.

An equal number of exploration and exploitation operators does not necessitate a balance between exploration and exploitation. The performance of operator is a critical factor.

Assessing the performance of an operator can be done by replacing the exploration or exploitation operator in one algorithm with the same type of operator in the other algorithm. Testing the modified algorithms with benchmark problems reveal whether or not the replaced operator was performing its function efficiently relative to the other operator.

To improve the performance of the algorithm such that the modified algorithm outperforms the two original algorithms, we aimed at using the best performing exploration and exploitation operators from the two algorithms. The hybrid algorithm, MAKHA, was constructed from the following processes:

- The watch-jump process.
- The foraging activity process.
- The physical random diffusion process.
- The genetic mutation and crossover process.
- The somersault process.

The climb process which consumes a high NFE was not included in hybrid algorithm nor was random diffusion step in most of the problems.

The general pseudo-code for this algorithm is shown in Fig. 3, while the equations used are as follows:

A. Initialization procedure

The positions of the hybrid agent (monkey/krill) are created randomly, $X_i = (X_{i1}, X_{i2}, ..., X_{iNV})$, where $i = 1$ to $NP$ which represents the number of hybrids, while $NV$ represents the dimension of the decision variable vector.

B. The fitness evaluation and sorting

$H_i = f(X_i)$, where $H$ stands for hybrid fitness and $f$ the objective function used.

C. The watch-jump process

In this process, each hybrid jumps with respect to other hybrids’ positions. The decision is based on information transferred by other hybrids as follows:

$$Y_{ij} = X_{ij} + \alpha_{ij}(X_{ij} - X_{kj}), \quad i = 1, 2, ..., NP$$

(1)

Full details about this step can be found in [15]

D. Foraging motion

Calculate the food attractiveness and the effect of best fitness so far

$$\beta_i^{food} = C_{food} \hat{H}_{i,food} \hat{X}_{i,food}$$

(2)

$$\beta_i^{best} = \hat{H}_{i,best} \hat{X}_{i,best}$$

(3)

Where $C_{food}$ is the food coefficient, which starts at 2 and decreases with time.
Part I: Set parameters and initialize
Define lower and upper bounds of X
Generate random population (NP) of positions (X_i)
Assign values to KHA parameters
Assign values to MA parameters
Set time, food foraging (Fi) and diffusion (Di) to zero
Calculate genetic parameters (Cr, Mu)

Part II: Algorithm loops
Evaluate the fitness of the hybrid individuals.
For I = 1: MG (Maximum Generation)
Watch-jump process
Generate new Y_i from (X_i-b, X_i+b)
If -f(Y_i) ≥ -f(X_i)
Update X_i with Y_i if feasible (i.e. within limits)
End If
Foraging motion step
Generate center of food density (X^food) for hybrids.
Calculate and update the foraging motion (F_i).
Physical diffusion step
Calculate and update the diffusion motion (D_i)
Move positions of hybrids after t+Δt period
Update the time (t)
Evaluate the fitness of the hybrid individuals (f(X))
Rank hybrid positions according to minimum fitness
Return the best hybrid position and its fitness so far
Implement the genetic operator
Apply crossover and mutation on hybrid positions
Update the hybrid positions and check their limits
Somersault process
Estimate Pivot (P) from X_i, c and d
Calculate Y_i around the pivot
Update X_i with Y_i if feasible
Repeat somersault until feasible
Rank the positions and find current best solution
End For MG
Get the best solution

Fig. 1. Pseudocode for MAKHA Algorithm

The center of food density is estimated from the shown equation:

\[ X^{food} = \frac{\sum_{i=1}^{NP} X_i}{\sum_{i=1}^{NP} H_i} \]  \hspace{1cm} (4)

and \( H_{\text{best}} \) is the best previously visited position.

\( \hat{H}_i, \hat{X}_i \) are unit normalized values obtained from this general form:

\[ \hat{X}_{i,j} = \frac{X_{i,j} - X_i}{\|X_{j} - X_i\|^2 + \varepsilon} \]  \hspace{1cm} (5)

\[ \hat{H}_{i,j} = \frac{H_i - \hat{H}_i}{H_{\text{worst}} - \hat{H}_i} \] \hspace{1cm} (6)

\( \varepsilon \) is a small positive number that is added to avoid singularities. \( H_{\text{best}} \) and \( H_{\text{worst}} \) are the best and the worst fitness values of the hybrid agents so far. H stands for the hybrid fitness.

The foraging motion is:

\[ F_i = V_f \beta_i + w_f F^\text{old}_i \] \hspace{1cm} (7)

Where \( V_f \) is the foraging speed, \( w_f \) is the inertia weight of the foraging motion in the range [0,1], and \( F^\text{old}_i \) is the last foraging motion.

E. Physical diffusion
This is an exploration step that is used at high dimensional problem.

\[ D_i = D_{\text{max}} \left(1 - \frac{I}{MG}\right) \delta \] \hspace{1cm} (8)

where \( D_{\text{max}} \) is the maximum diffusion speed, and \( \delta \) is the random direction vector. Then, the time interval \( \Delta t \) is calculated from

\[ \Delta t = C_t \sum_{L=1}^{NV} (UB_L - LB_L) \] \hspace{1cm} (9)

where \( C_t \) is constant. The step for position is calculated from

\[ \frac{dx_i}{dt} = F_i + D_i \] \hspace{1cm} (10)

\[ X_i(t + \Delta t) = X_i(t) + \Delta t \frac{dx_i}{dt} \] \hspace{1cm} (11)

where \( \frac{dx_i}{dt} \) represents the velocity of hybrid.

F. Genetic Operator
The genetic operator is implemented through two steps, crossover and mutation:

1) Crossover
\[ X_{i,m} = \begin{cases} X_{p,m}, & \text{random} < C_r \\ X_{i,m}, & \text{otherwise} \end{cases} \] \hspace{1cm} (12)

where \( C_r \) is the crossover probability.

2) Mutation
\[ X_{i,m} = \begin{cases} X_{g\text{best},m} + \mu(X_{p,m} - X_{a,m}), & \text{random} < Mu \\ X_{i,m} \end{cases} \] \hspace{1cm} (13)

where \( \mu \) is a random number, and \( Mu \) is the mutation probability.
G. The somersault process:

\( \alpha \) is generated randomly from \([c, d]\) where \(c\) and \(d\) are somersault interval. The process is implemented as follows

Create a pivot \( P_i \):

\[
P_i = \frac{1}{NF - 1} \sum_{j=1}^{NF} (\sum_{i=1}^{NF} X_{ij} - X_{ij}) \tag{14}
\]

Get Hybrid new position, \( Y \)

\[
Y_i = X_i + \alpha |P_i - X_{ij}| \tag{15}
\]

MAKHA offers more exploration than KHA and more exploitation than MA.

III. DESCRIPTION OF THE PHASE EQUILIBRIUM PROBLEMS

In this study, the phase stability and equilibrium problems are stated as a global optimization problem. Therefore, the global optimization problem to be solved is: Minimize \( F(X) \) with respect to \( D \) decision variables: \( X = (X_1, \ldots, X_D) \). The upper and lower bounds of these variables are \( a \) and \( b \), respectively. The phase stability, phase equilibrium and reactive phase equilibrium calculations for testing the performance of global optimization methods are explained below.

A. Phase Stability

It involves the determination of whether a system will remain in one phase at the given conditions or split into two or more phases. The thermodynamics function is the tangent plane distance function

\[
TPDF = \sum_{i=1}^{c} y_i \left( \mu_i^y - \mu_i^x \right) \tag{16}
\]

where \(c\) is the number of components of the mixture, \( \mu_i^y \) and \( \mu_i^x \) are the chemical potentials calculated at trial composition \( y \) and feed composition \( x \). The optimization problem is to minimize (15) and the decision variables, \( \beta \) are given by

\[
n_{iy} = \beta_i z_i n_F \quad i = 1, \ldots, c \tag{17}
\]

\[
y_j = \sum_{i=1}^{c} n_{iy} \quad i = 1, \ldots, c \tag{18}
\]

where \(n_{iy}\) are the mole numbers of component \(i\) in phase \(y\) and \(nF\) is the total moles in the mixture under analysis.

B. Phase Equilibrium Calculations

It involves the determination of the number, type and composition of the phases at equilibrium at the given operating conditions. The thermodynamic function is the Gibbs energy of mixing given below

\[
g = \sum_{j=1}^{\pi} \sum_{i=1}^{c} n_{ij} \ln(x_{ij}y_{ij}) = 
\sum_{j=1}^{\pi} \sum_{i=1}^{c} n_{ij} \ln \left( \frac{x_{ij} \phi_{ij}}{\phi_i} \right) \tag{19}
\]

where \(\pi\) is the number of phases at equilibrium and \(\theta_{ij}\) denotes the composition (i.e., \(x\) or \(n\)) or thermodynamic property of component \(i\) in phase \(j\). The optimization problem is to minimize (18) with respect to the decision variables \(\beta\)

\[
n_{ij} = \beta_{ij} z_{ij} n_F \quad i = 1, \ldots, c \tag{20}
\]

\[
n_{ij} = \beta_{ij} z_{ij} n_F \quad i = 1, \ldots, c \tag{21}
\]

\[
n_{i\alpha} = z_i n_F - \sum_{m=1}^{\pi-1} n_{im} \quad i = 1, \ldots, c \tag{22}
\]

C. Reactive Phase Equilibrium Calculations

It involves the determination of the number, type and composition of the phases at equilibrium at the given operating conditions and subject to element/mass balances and chemical equilibrium constraints. Gibbs free energy of mixing defined using reaction equilibrium constants \([2]\)

\[
G_K = g - \sum_{j=1}^{\pi} \ln K_{eq} N^{-1} n_{ref,j} \tag{23}
\]

where \(g\) is the Gibbs free energy of mixing. \(\ln K_{eq}\) is a row vector of logarithms of chemical equilibrium constants for \(r\) independent reactions. \(N\) is an invertible, square matrix formed from the stoichiometric coefficients of a set of reference components chosen from the \(r\) reactions, and \(n_{ref}\) is a column vector of moles of each of the reference components. The optimization problem is to minimize \(G_K\) subject to

\[
\sum_{j=1}^{r} \left( n_{ij} - v_i N^{-1} n_{ref,j} \right) = n_{iF} - v_i N^{-1} n_{ref,F} \tag{24}
\]

where \(n_{iF}\) is the initial moles of component \(i\) in the feed, \(v_i\) is the row vector (of dimension \(r\)) of stoichiometric coefficients of component \(i\) in \(r\) reactions, and \(n_{iF}\) is the number of moles of component \(i\) in phase \(j\). The constrained global optimization problem can be solved by minimizing \(G_K\) with respect to \(c\) \((\pi-1) + r\) decision variables \(n_{ij}\). In this formulation, the mass balance equations are rearranged to reduce the number of decision variables of the optimization problem and to eliminate equality constraints.

For interested readers, several references provide a detailed description of these thermodynamic calculations \([2,4,6]\). Previous work reported the evaluation of global optimization methods for solving twenty-four problems \([5,6,8]\). In this work, we focused on the nine most-difficult ones. The basis for the selection was the relatively lower success rates that optimization methods obtained when solving them in the previous studies. These problems are presented in Table I.
The maximum and then stopped. This maximum value for the number of iterations was different for different algorithms. Calculations were performed for a certain number of iterations to determine the reliability of the optimization algorithms. Parameters were tuned using preliminary calculations and are shown in Table II. Each problem was solved 30 times independently and with different random initial seeds to show the reliability of the optimization algorithms.

**Details of numerical implementation and performance metrics used for testing the algorithms**

All thermodynamic problems and the different optimization algorithms were coded in the MATLAB® technical computing environment. MAKHA was developed and coded by the authors. The code for CS was obtained from MATLAB file exchange server as uploaded by their developers and used as is. Parameters were tuned using preliminary calculations and are shown in Table II. Each problem was solved 30 times independently and with different random initial seeds to determine the reliability of the optimization algorithms. Calculations were performed for a certain number of iterations and then stopped. This maximum value for the number of iterations was different for different algorithms. The maximum values were selected to give the same number of function evaluations at the end of the run.

The methods were evaluated according to the reliability and efficiency for finding the global optimum. The efficiency is determined by recording the number of function evaluations NFE for each optimization algorithm, where a low value of NFE means a higher efficiency. Note that NFE is an unbiased indicator of the computational costs required by a certain algorithm and is independent on the host hardware. In this work, we present a different reliability metric: a plot of the average best value against the number of function evaluations against NFE, which is calculated at each iteration. Since the NF needed for each iteration differ amongst the optimization methods, the plot of average best value against NFE is a good indicator of the computational costs required by a certain algorithm.

**TABLE I. DETAILS OF THE PHASE STABILITY, PHASE EQUILIBRIUM AND REACTIVE PHASE EQUILIBRIUM PROBLEMS USED IN THIS STUDY**

<table>
<thead>
<tr>
<th>Code</th>
<th>System</th>
<th>Feed conditions</th>
<th>Thermodynamic models</th>
<th>Global optimum</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>T7</strong></td>
<td>$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_{7,16} + C_{17,4}$</td>
<td>$n_p = (0.7212, 0.09205, 0.04455, 0.03123, 0.01273, 0.01361, 0.07215, 0.01248)$ at 353K and 38500kPa</td>
<td>Phase stability problem with SRK EoS with classical mixing rules.</td>
<td>-0.002688</td>
</tr>
<tr>
<td><strong>T8</strong></td>
<td>$C_1 + C_2 + C_3 + iC_4 + C_4 + iC_5 + C_5 + C_6 + iC_15$</td>
<td>$n_p = (0.614, 0.10259, 0.04985, 0.008989, 0.02116, 0.00722, 0.01187, 0.01435, 0.16998)$ at 314K and 2010.288kPa</td>
<td>Phase stability problem with SRK EoS with classical mixing rules.</td>
<td>-1.486205</td>
</tr>
<tr>
<td><strong>T9</strong></td>
<td>$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7 + C_8 + C_9 + C_{10}$</td>
<td>$n_p = (0.6436, 0.0752, 0.0474, 0.0412, 0.0297, 0.0138, 0.0303, 0.0371, 0.0415, 0.0402)$ at 435.35K and 19150kPa</td>
<td>Phase stability problem with SRK EoS with classical mixing rules.</td>
<td>-0.0000205</td>
</tr>
<tr>
<td><strong>G4</strong></td>
<td>$C_1 + H_2S$</td>
<td>$n_p = (0.9813, 0.0187)$ at 190K and 4053kPa</td>
<td>Phase equilibrium problem with SRK EoS with classical mixing rules.</td>
<td>-0.019892</td>
</tr>
<tr>
<td><strong>G6</strong></td>
<td>$C_2 + C_3 + C_4 + C_5 + C_6$</td>
<td>$n_p = (0.401, 0.293, 0.199, 0.0707, 0.0363)$ at 390K and 5583kPa</td>
<td>Phase equilibrium problem with SRK EoS with classical mixing rules.</td>
<td>-1.183653</td>
</tr>
<tr>
<td><strong>G7</strong></td>
<td>$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_{7,16} + C_{17,4}$</td>
<td>$n_p = (0.7212, 0.09205, 0.04455, 0.03123, 0.01273, 0.01361, 0.07215, 0.01248)$ at 353K and 38500kPa</td>
<td>Phase equilibrium problem with SRK EoS with classical mixing rules.</td>
<td>-0.838783</td>
</tr>
<tr>
<td><strong>G8</strong></td>
<td>$C_1 + C_2 + C_3 + iC_4 + C_4 + iC_5 + C_5 + C_6 + iC_15$</td>
<td>$n_p = (0.614, 0.10259, 0.04985, 0.008989, 0.02116, 0.00722, 0.01187, 0.01435, 0.16998)$ at 314K and 2010.288kPa</td>
<td>Phase equilibrium problem with SRK EoS with classical mixing rules.</td>
<td>-0.769772</td>
</tr>
<tr>
<td><strong>R4</strong></td>
<td>$A_1 + A_2 \leftrightarrow A_3 + A_4$</td>
<td>$n_p = (0.3, 0.4, 0.3, 0.0)$ at 298.15K and 101.325kPa</td>
<td>Reactive phase equilibrium problem with UNIQUAC model and ideal gas. $lnK_{eq} = 450/T + 0.8$</td>
<td>-1.10630</td>
</tr>
<tr>
<td><strong>R7</strong></td>
<td>$A_1 + A_2 \leftrightarrow A_3$</td>
<td>$n_p = (0.52, 0.48, 0.0)$ at 323.15K and 101.325kPa</td>
<td>Reactive phase equilibrium problem with Margules solution model. $K_{eq} = 3.5$</td>
<td>-0.653756</td>
</tr>
</tbody>
</table>

**D. Details of numerical implementation and performance metrics used for testing the algorithms**

All thermodynamic problems and the different optimization algorithms were coded in the MATLAB® technical computing environment. MAKHA was developed and coded by the authors. The code for CS was obtained from MATLAB file exchange server as uploaded by their developers and used as is. Parameters were tuned using preliminary calculations and are shown in Table II. Each problem was solved 30 times independently and with different random initial seeds to determine the reliability of the optimization algorithms. Calculations were performed for a certain number of iterations and then stopped. This maximum value for the number of iterations was different for different algorithms. The maximum values were selected to give the same number of function evaluations at the end of the run.
indication of reliability versus efficiency of the optimization method.

### TABLE II. PARAMETERS FOR THE STOCHASTIC METHODS

<table>
<thead>
<tr>
<th>Method</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>c</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>d</td>
<td>-1</td>
</tr>
<tr>
<td>MAKHA</td>
<td>D\text{max} &amp; [0.001,0.02]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C_i</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>V_f</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>W_i</td>
<td>0.1</td>
</tr>
<tr>
<td>KHA</td>
<td>D\text{max} &amp; 0.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C_i</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>V_f</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>N\text{max} &amp; 0.01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W_i    &amp; [0.1,0.9]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W_d    &amp; [0.1,0.9]</td>
<td></td>
</tr>
<tr>
<td>MA</td>
<td>a</td>
<td>0.00001</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>d</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>N_c</td>
<td>10</td>
</tr>
<tr>
<td>CS</td>
<td>p</td>
<td>0.25</td>
</tr>
</tbody>
</table>

### IV. RESULTS AND DISCUSSION

The results are presented in two different ways. For each problem, the mean best values are plotted versus NFE for each of the four algorithms. Three of these plots, one for each problem category, are shown here as representatives of the results. The minimum NFE required to reach a certain tolerance from the known global minimum for each problem were calculated and presented in Table III. A detailed discussion of the results follows.

#### A. Phase stability problems

Problem T7 is a nine-variable phase-stability problem that is extremely difficult to solve. The means of the minimum values obtained by all methods were not close enough to the global minimum except for CS, which was able to find the global minimum down to a tolerance of 10^{-4}. MAKHA outperformed both MA and KHA and was able to find the global minimum down to a tolerance of 10^{-5}. Fig. 2 shows how the two original problems, MA and KHA, were trapped in a local minimum and was unable to find its global minimum. However, none of the algorithms was able to get closer to the global minimum within the total NFE run in our study, as illustrated by the NFE values of Table III.

Problem T8 is also a difficult phase-stability problem. KHA outperformed both MAKHA and CS in solving this problem down to a tolerance of 10^{-5}. However, none of the algorithms was able to get closer to the global minimum within the total NFE run in our study, as illustrated by the NFE values of Table III.

Problem T9 is the last of the three phase stability problems. Even though, KHA was again the most efficient algorithm down to a tolerance of 10^{-5} but it failed to go any further. MAKHA was more reliable down to 10^{-6} and CS was the most reliable to down to 10^{-7} tolerance level.

For the phase stability problems, CS is clearly the most reliable method. It may not be as efficient as making the global minimum as other methods but it outperforms the rest in terms of finding the global minimum. However, the hybridization of MA and KHA resulted in an improvement in reliability as MAKHA was more reliable than the two original algorithms.

#### b) Phase equilibrium problems

Problem G4 is a two-variable phase equilibrium problem that is relatively easy to solve. However, KHA seemed to have been trapped in a local minimum and was unable to find the global minimum, within a tolerance of 10^{-3}, as shown in Figure 3. MA performed better than KHA but it was inefficient and was unable to reach the global minimum at the 10^{-7} level. MAKHA and CS were reliable and efficient in solving this problem.

Despite the fact that KHA was not able to solve adequately problem G4, it was superior in solving problem G6 down to 10^{-7} tolerance level. It was not able to solve it any further to the end of the runs. Again, MAKHA and CS were more reliable in solving this problem. For Problem G7, MAKHA again was more reliable than the two original algorithms. G8, on the other hand, was the only problem that MAKHA was less reliable than KHA in its solution. MA was not able to reach the global minimum at any tolerance level. CS was the most reliable problem but less efficient than KHA.

The convergence profiles of the four phase equilibrium problems (G4, G6, G7 and G8) indicated that CS is the most reliable of all algorithms as it was the only one to be able to solve all problems down to the 10^{-7} tolerance level. MAKHA has shown better reliability than the two original algorithms but less efficient than KHA at the higher tolerance levels.

#### c) Reactive phase equilibrium problems

Regardless of the number of variables, the reactive phase equilibrium problems are more difficult than the non-reactive phase equilibrium problems, because the chemical reaction equilibrium constraints must be satisfied. Problem R4 was successfully solved down to the 10^{-5} tolerance level by CS, which was also able to converge to the global minimum at the 10^{-7} level. MAKHA performed better than MA and KHA but all of them were not able to arrive even at a level of 10^{-4} from the global minimum.

Problem R7 is a good indication of the significance of this study. Even though both MA and KHA failed completely in solving this problem, MAKHA was able to converge to the global minimum down to the 10^{-7} tolerance level. Its performance was similar to that of CS as shown in Figure 4.

#### V. CONCLUSIONS

In this study, we have developed a hybrid optimization algorithm that combines feature from MA and KHA. The developed algorithm was used to solve nine difficult phase stability and phase equilibrium problems. Its performance was compared against the two original algorithms and CS, which is considered the most reliable algorithm for solving this type of problems. These thermodynamic problems were systematically solved by the proposed hybrid algorithm.
solved by the different metaheuristics and the results were tracked and compared. The results clearly show that MAKHA is more reliable than the original two algorithms. Even though CS is still the most reliable of all tested optimization methods, MAKHA’s performance was close second.

REFERENCES

TABLE III. Minimum NFE for the average best value to reach 1E-3, 1E-4, 1E-5, 1E-6, and 1E-7 from the known global minimum

<table>
<thead>
<tr>
<th>Metaheuristic</th>
<th>g</th>
<th>T7</th>
<th>T8</th>
<th>T9</th>
<th>G4</th>
<th>G6</th>
<th>G7</th>
<th>G8</th>
<th>R4</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MAKH</strong></td>
<td>1E-3</td>
<td>34222</td>
<td>4356</td>
<td>602</td>
<td>61</td>
<td>302</td>
<td>482</td>
<td>94579</td>
<td>53152</td>
<td>21659</td>
</tr>
<tr>
<td></td>
<td>1E-4</td>
<td>81217</td>
<td>91869</td>
<td>1505</td>
<td>61</td>
<td>453</td>
<td>6266</td>
<td>140649</td>
<td>(\infty)</td>
<td>56507</td>
</tr>
<tr>
<td></td>
<td>1E-5</td>
<td>167495</td>
<td>148237</td>
<td>33712</td>
<td>1037</td>
<td>604</td>
<td>81458</td>
<td>168562</td>
<td>(\infty)</td>
<td>62799</td>
</tr>
<tr>
<td></td>
<td>1E-6</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>140868</td>
<td>5002</td>
<td>1057</td>
<td>117126</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>64372</td>
</tr>
<tr>
<td></td>
<td>1E-7</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>9150</td>
<td>52246</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>82159</td>
<td></td>
</tr>
<tr>
<td><strong>KHA</strong></td>
<td>1E-3</td>
<td>(\infty)</td>
<td>1000</td>
<td>302</td>
<td>41</td>
<td>152</td>
<td>242</td>
<td>1364</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-4</td>
<td>(\infty)</td>
<td>1364</td>
<td>706</td>
<td>41</td>
<td>305</td>
<td>1376</td>
<td>2365</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-5</td>
<td>(\infty)</td>
<td>2183</td>
<td>4847</td>
<td>(\infty)</td>
<td>509</td>
<td>48599</td>
<td>3184</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-6</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>917</td>
<td>(\infty)</td>
<td>5004</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-7</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>25843</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td><strong>MA</strong></td>
<td>1E-3</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>82000</td>
<td>820</td>
<td>6150</td>
<td>13120</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-4</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>274700</td>
<td>820</td>
<td>12300</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-5</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>14760</td>
<td>22550</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-6</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>70520</td>
<td>38950</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td></td>
<td>1E-7</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
<td>(\infty)</td>
</tr>
<tr>
<td><strong>CS</strong></td>
<td>1E-3</td>
<td>41120</td>
<td>17820</td>
<td>7400</td>
<td>120</td>
<td>1700</td>
<td>7840</td>
<td>21780</td>
<td>16100</td>
<td>11120</td>
</tr>
<tr>
<td></td>
<td>1E-4</td>
<td>84000</td>
<td>35460</td>
<td>11400</td>
<td>120</td>
<td>3700</td>
<td>25440</td>
<td>45540</td>
<td>32700</td>
<td>20400</td>
</tr>
<tr>
<td></td>
<td>1E-5</td>
<td>113760</td>
<td>60960</td>
<td>59400</td>
<td>1240</td>
<td>5100</td>
<td>54240</td>
<td>76860</td>
<td>52700</td>
<td>30640</td>
</tr>
<tr>
<td></td>
<td>1E-6</td>
<td>135520</td>
<td>(\infty)</td>
<td>95400</td>
<td>2760</td>
<td>6900</td>
<td>96800</td>
<td>111060</td>
<td>71700</td>
<td>41200</td>
</tr>
<tr>
<td></td>
<td>1E-7</td>
<td>157920</td>
<td>(\infty)</td>
<td>135000</td>
<td>4120</td>
<td>24700</td>
<td>144480</td>
<td>151740</td>
<td>90300</td>
<td>53040</td>
</tr>
</tbody>
</table>

Fig. 2. The evolution of the mean best value calculated via the four metaheuristics versus NFE for problem T7
Fig. 3. The evolution of the mean best value calculated via the four metaheuristics versus NFE for problem G4

Fig. 4. The evolution of the mean best value calculated via the four metaheuristics versus NFE for problem R7