A Memetic Algorithm Applied to the Optimal Design of a Planar Mechanism for Trajectory Tracking

Eduardo Vega-Alvarado, Edgar Alfredo Portilla-Flores, Efrén Mezura-Montes,
Leticia Flores-Pulido, Maria Bárbara Calva-Yáñez

***Abstract*—Memetic algorithms (MA), explored in recent literature, are hybrid metaheuristics formed by the synergistic combination of a population-based global search technique with one or more local search algorithms, which in turn can be exact or stochastic methods. Different versions of MAs have been developed, and although their use was focused originally on combinatorial optimization, nowadays there are memetic developments to solve a wide selection of numerical type problems: with or without constraints, mono or multi objective, static or dynamic, among others. This paper presents the design and application of a novel memetic algorithm, MemMABC, tested in a case study for optimizing the synthesis of a four-bar mechanism that follows a specific linear trajectory. The proposed method is based on the MABC algorithm as a global searcher, with the addition of a modified Random Walk as a local searcher. MABC is a modified version of the Artificial Bee Colony algorithm, adapted to handle design constraints by implementing the feasibility rules of Deb. Four-bar mechanisms are a good example of hard optimization problems, since they are used in a wide variety of industrial applications; simulation results show a high-precision control of the proposed trajectory for the designed mechanism, thus demonstrating that MemMABC can be applied successfully as a tool for solving real-world optimization cases.**

***Index Terms*—Four-bar mechanism, hard optimization, memetic algorithm, random walk.**

I. Introduction

METAHEURISTICS are algorithms designed to solve a wide variety of hard optimization problems in an approximate way, using trial and error techniques. The general characteristics of a metaheuristic are: it is inspired on natural or artificial processes, uses stochastic components (involving random variables), and has a series of parameters that must be adjusted to the specific case [1].

Manuscript received on March 06, 2016, accepted for publication on June 20, 2016, published on June 25, 2016.

Eduardo Vega-Alvarado is with Universidad Autónoma de Tlaxcala, Facultad de Ciencias Básicas, Ingeniería y Tecnología and with Instituto Politécnico Nacional, Centro de Innovación y Desarrollo Tecnológico en Cómputo, Mexico (e-mail: evega@ipn.mx).

Leticia Flores-Pulido is with Universidad Autónoma de Tlaxcala, Facultad de Ciencias Básicas, Ingeniería y Tecnología, Mexico (e-mail: aicitel.flores@gmail.com).

Edgar Alfredo Portilla-Flores and Maria Bárbara Calva-Yáñez are with Instituto Politécnico Nacional, Centro de Innovación y Desarrollo Tecnológico en Cómputo, Mexico (e-mail: aportilla@ipn.mx, b_calva@hotmail.com).

Efrén Mezura Montes is with Universidad Veracruzana, Centro de Investigación en Inteligencia Artificial, Mexico (e-mail: emezura@uv.mx).

Population-based metaheuristics start from an initial set of solutions or proposed individuals to find the optimal value or values of a problem, and there are two general groups of these algorithms: evolutive computing and swarm intelligence. The two main tasks of modern metaheuristics are diversification (exploration) and intensification (exploitation) [2]. Techniques based on population are good for exploring, but usually are deficient when exploiting [1]. Several alternatives have been developed in order to solve this weakness, and Memetic Algorithms (MAs) highlight between them for synergically combining the global search dynamics of a population metaheuristic with the refinements of a local search (LS), in order to obtain a hybrid method of solution [3], [4].

The first MAs were taken with suspicion because of their metaheuristic nature, but in the last decade these techniques have been applied successfully to solve different problems including numerical cases with constraints, and dynamic multi-objective optimization [3], [5]–[7]. In the real world, they have been used in security and cryptanalysis [8], control systems [9], task scheduling and routing [10], and data classification [11], among others.

In mechanical engineering, *synthesis* is the design process of machines or mechanical systems [12]. The purpose of the mechanism determines the type of synthesis to carry out: generation of motion, function or trajectory. In the synthesis for function generation, an input motion to the mechanism is correlated with another for output; in trajectory generation, a point is controlled to track a line in a plane, such that it assumes a prescribed set of sequential positions [13]. This work addresses the dimensional synthesis of a mechanism: calculate the length of the necessary links for tracking a specific trajectory.

The four-bar mechanism has been widely used in engineering design, since it is the simplest articulated mechanism for controlled movement with a degree of freedom. The synthesis of four-bar mechanisms for trajectory tracking is a well-known numerical problem previously explored in depth, and classical approaches have been used for this synthesis, including graphical and analytical methods; however, they have a limitation regarding the number of points to be tracked, since solutions are extremely complicated for problems with more than four points. For this reason, the design of these mechanisms is a typical case of hard numerical optimization.

Hard optimization problems can't be solved in an optimal way or to a guaranteed limit using deterministic methods and with normal computing resources. Taking into account that most real-world optimization cases are hard problems, it is necessary to develop alternative methods for their solution. In the search of new ideas for improving the performance of metaheuristics several models have been implemented, based on natural or artificial processes.

In this work, the design of a new memetic algorithm (MemMABC) is presented, with its application to the optimal synthesis of a four-bar mechanism for the control of a trajectory delimited by a set of N precision points tracked by the coupler. MemMABC is a combination of MABC and a version of the Random Walk algorithm (RW) modified for handling design constraints by implementing the feasibility rules of Deb. Although these building blocks are well-known methods usually they are applied only in an individual way; this is a novel approach to solve constrained problems of numerical optimization by combining synergically two strategies of different nature, applied to a real-world engineering case.

The paper is organized as follows: Section 2 shows the basic model of memetic algorithms and the implementation of local search stages. In Section 3, the MemMABC algorithm is introduced, with an analysis of both its global and local searchers. Section 4 describes the problem of mechanism synthesis, with a brief explanation of the kinematics. A case study with a specific optimization problem is analyzed in Section 5, including the description of the design variables. Section 6 presents the applied algorithm, with special emphasis on its computational implementation. Finally, experimental results are reported in Section 7, while the conclusions of this paper are included in Section 8.

II. Memetic Algorithms

Early work on MAs dates from the 80s [1], [14]; as a consequence of the strengthening in evolutive algorithms, new ideas were implemented in order to improve their performance, once that their limitations were known. In 1989, Moscato [15] proposed the memetic algorithms, to simulate the cultural evolution process derived from Lamarck's evolution theory and the *meme*, presented by R. Dawkins as the equivalent to the gene in natural evolution. MAs constitute a general method based on the sinergetic combination of algorithms for global and local search, in a new optimization philosophy [16]. A meme corresponds to recurrent patterns in real world or to specific knowledge, and is coded for the effective solution of problems as the building block of cultural know-how that is transmissible and reproducible [17], [18].

Three stages can be identified in the evolution of MAs [4]:

1) Applications are made by simple combination of an evolutive algorithm and a specialized method of local search.
2) Multiple memes are used, and the developments include any population-based algorithm as a global search tool.
3) Explicit mechanisms of learning are incorporated (adaptive MAs), with the use of exact methods in tandem configuration for local search.

MAs of first and second stages are still being developed because of their simplicity and efficiency, with an improvement over the performance of previously applied single metaheuristics.

A. Basic Model of a Memetic Algorithm

Figure 1 shows the block diagram of a basic population metaheuristic, indicating the four points where a local search can be included in order to form a MA [19], [20]:

1) On the population, to simulate the cultural development that will be transmitted from one generation to another; it can be applied to the whole set of agents or to specific elements, and even to the initial group.
2) On the parent or selected parents, before reproduction stage.
3) When new solutions are generated, to produce a better offspring.
4) On the offspring, before selecting a survivor according to fitness criteria.

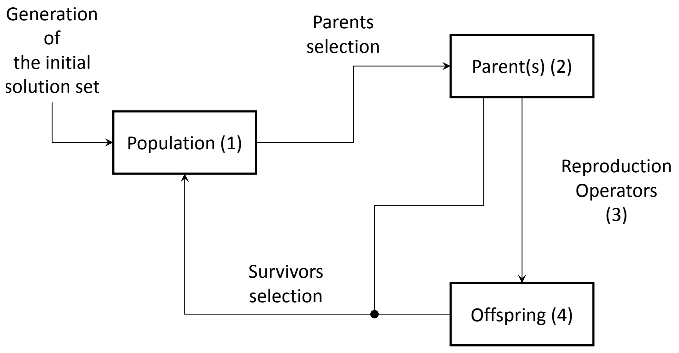

Fig. 1. Block diagram of a basic population metaheuristic

From this basic model several versions of MAs have been developed, differing between each other in at least one of the following aspects:

– Population metaheuristic used as a base.
– Selected algorithm for local search (exact method or metaheuristic, number of memes to consider.)
– Conditions for local search (trigger event, frequency, intensity, number of individuals to improve, etc.) [18], [21].

Originally MAs were based on evolutive algorithms but nowadays their implementation with swarm intelligence methods is common, using algorithms such as Particle Swarm Optimization (PSO) [11], Artificial Bee Colony (ABC) [22], Harmony Search (HS) [23], and Fire Fly (FF) [24].

B. Local Search

Local search algorithms are stochastic or deterministic methods that take as an input a solution generated randomly or by a specific algorithm, looking for transitions with the neighbors to this point at a given time. The goal is to find a better individual and to convert it in the next configuration, maintaining the original element if no improvement is detected [25]. The concept of vicinity is fundamental for LS since it represents the search area for individual refinement; in the case of combinatorial optimization this area is formed by the set of all solutions that can be reached by a unitary change in the current individual, while in continuous or numerical problems it is a dense set formed by an infinite number of points, and a modification strategy is required to find the neighbors [20].

Finding a good balance between the components of global and local search is one of the main design goals in a MA, and can be seen as an optimization process *per se*. From the way a memetic algorithm is compounded and in the implementation of its LS, it is seen that the equality MA = GS + LS is incorrect, since both searches are interrelated and are not designed as independent stages [14].

C. Classification of MAs

A MA can be classified as simple (canonical) or adaptive [26]. Simple MAs are characterized for *a priori* knowledge of the problem domain, that is incorporated to the algorithm design and produces a static behavior; in spite of belonging to the first generation of MAs, canonical hybrids are still popular for their easy implementation, particularly because of the use of genetic algorithms as global searchers.

On the other hand, adaptive MAs acquire information during their execution (learning), so they are able to reconfigure not only their parameters but their operators at run time, in order to adapt themselves to specific circumstances or instances of the problem [18]. The design of an adaptive MA requires to consider such aspects as the selection of subsets with agents for applying the fine search, the frequency and intensity of LS stages, the selection of procedures for the improvement, and the convergence of the population [6], [22].

III. MemMABC Algorithm

The Artificial Bee Colony is a swarm intelligence algorithm introduced by Karaboga as a method for numerical optimization [27], inspired on the behavior of bee hives in two natural processes: the recruitment of bees for the exploitation of food sources and the abandonment of exhausted sources. In ABC, the bees in a hive are divided in three groups: employed, onlookers and scouts, and each group represents an evaluation stage. There is an employed bee assigned to each source, and from this point the bee calculates a new solution and keeps the best of both. The number of onlookers is the same as the employed bees, and their assignation to sources is determined by the performance of such sources. The onlookers also calculate new solutions from their assigned source. Finally, when a source can't improve after a specified number of cycles, it is abandoned and replaced by a new one found by a scout bee.

Several versions of ABC have been developed; the modification proposed in [28], MABC, has an adaptation for constrained numerical optimization with a tournament-type selection based on the feasibility rules of Deb [29]. These criteria improve the process for selecting solutions at each iteration, by choosing the most feasible individual instead of the one with the best value for the objective function:

1) Between two feasible solutions, the best objective function value is preferred.
2) Between a feasible solution and another infeasible, the feasible is selected.
3) Between two infeasible solutions, the lowest sum of constraint violations (CVS) is preferred.

In this development a novel canonical MA, MemMABC, was designed taking as a base the MABC algorithm for global searching, with a modification to include a LS activated by time; Algorithm 1, *A1*, shows this memetic. The trigger for the LS stages is controlled by the variable $Frequency$, which indicates the period between an event of LS and the next one, in terms of a number of cycles or generations (line 39, A1). Although the iterations in original ABC and MABC are controlled by the number of cycles, MemMABC uses a variation to stop after a fixed number of objective function evaluations. This implementation permits a fair comparison [30] of MemMABC with other algorithms, specifically with MABC for the purposes of this work.

Algorithm 2, *A2*, shows the implementation of the local search method in MemMABC. It is a version of RW, modified to handle constraints by implementing the rules of Deb (line 21, A2). RW was chosen because of their easy implementation, since this method does not require the derivative of the objective function to calculate its gradient or its Hessian. Another modification was introduced to the original RW in order to reduce the computing effort and execution time, taking into account the complexity and high dimensionality of some real-world problems. RW requires the random generation of a number set R with values in the interval [-1,1], whose cardinality corresponds to the number of design variables, n. The values in R are transformed into search directions, so it is necessary to avoid a bias toward the diagonals of the unit hypercube surrounding the initial search point [31]. The random numbers generated are accepted only if $R < 1$, with R being computed as

$$R = (r_1^2 + r_2^2 + ... + r_n^2)^{1/2} \tag{1}$$

But when the problem to solve has a high dimensionality the algorithm can take several iterations to find a valid combination. This can be avoided if the elements on R are downsized when generated, using a constant divider (line 10, A2). The resultant array is a subset of R, so it is a valid combination. Finally, the LS depth or intensity is controlled by

Algorithm 1. MemMABC

```
begin
    set $Frequency, MaxEvs$ ;
    $Evaluations = 0, g = 1$;
    initialize the set of food sources $x_i^0, i = 1, ..., SN$;
    evaluate CVS and objective function for
    $x_i^0, i = 1, ..., SN$;
    if there are equality constraints then initialize $\epsilon(g)$;
    repeat
        if there are equality constraints then
            evaluate each $x_i^0, i = 1, ..., SN$ with $\epsilon(g)$;
        end
        for $I = 1$ to $SN$ do
            generate $v_i^g$ with $x_i^{g-1}$;
            evaluate CVS and objective function for $v_i^g$;
            $Evaluations = Evaluations + 1$ ;
            if $v_i^g$ is better $x_i^{g-1}$ (Deb's criteria) then
                $x_i^g = v_i^g$;
            else
                $x_i^g$=$x_i^{g-1}$;
            end
        end
        for $I = 1$ to $SN$ do
            select food source $x_i^g$ based on binary
            tournament selection;
            generate $v_i^g$ with $x_i^g$;
            evaluate CVS and objective function for $v_i^g$;
            $Evaluations = Evaluations + 1$ ;
            if $v_i^g$ is better than $x_i^{g-1}$ (Deb's criteria) then
                $x_i^g = v_i^g$;
            end
        end
        apply smart flight to those solutions whose limit
        to be improved has been reached;
        make $Evaluations = Evaluations + 1$ for
        every source improved ;
        keep the best solution so far $x(Best)^g$;
        $g = g + 1$;
        if there are equality constraints then update $\epsilon(g)$
        if $((g)mod(Frequency)) == 0$ then
            apply RW to $x(Best)^g$;
            $TryLimit(Best)^g = 0$;
        end
    until $Evaluations >= MaxEvs$;
end
```

the parameter $MaxEvs$, who indicates the maximum number of evaluations for each activation.

IV. Analysis of the Four-bar Mechanism

A planar four-bar mechanism is formed by a reference bar (r_1), a crank or input bar (r_2), a coupler (r_3), and a rocker

Algorithm 2. Local search using RW

```
begin
    set $MaxIter, MaxCount$;
    take best solution so far as initial point $X_0$;
    evaluate CVS and objective function for $X_0$;
    $Iterations = 1$, $Go = True$, $Evaluations = 1$;
    while Go==True do
        repeat
            $R = 0$;
            for $J = 1$ to $Var$ do
                $Dir(J) = -0.1 + 2 * rand(1, Var)/10$;
                $R = R + Dir(J)^2$;
            end
            $R = sqrt(R)$;
        until $R < 1$;
        for $J = 1$ to $Var$ do
            $U(J) = Dir(J)/R$;
            $X_{Try}(J) = X1(J) + \lambda * U(J)$;
        end
        evaluate SVR and objective function for $X_{Try}$;
        $Evaluations = Evaluations + 1$ ;
        select best from $X_{Try}$ and $X_0$ using Deb's
        criteria;
        if $MaxIter < Iterations$ then
            $\lambda = \lambda/2$;
            $Iterations = 1$;
            if $\lambda <= \epsilon$ then
                $Go = False$;
            end
        else
            $Iterations = Iterations + 1$;
        end
        if $Evaluations >= MaxCount$ then
            $Go = False$;
        end
    end
end
```

or output bar (r_4), as is shown in Figure 2. Two coordinate systems are proposed in order to analyze this mechanism: a system fixed to the real world (OXY) and another for self reference (OX_rY_r), where (x_0, y_0) is the distance between the origin of both systems, θ_0 is the rotation angle of the reference system and θ_i ($i = 2, 3, 4$) corresponds to the angles for the bars in the mechanism; finally, the coordinate pair (r_{cx}, r_{cy}) indicates the length of the support bars to position the coupler C.

A. Kinematics of the mechanism

The kinematics of four-bar mechanisms have been extensively treated, detailed explanations are in [32] and [33]. For analyzing the mechanism position the closed loop equation

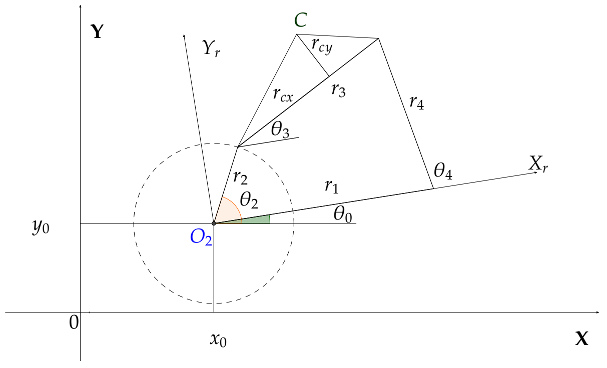

Fig. 2. Four-bar mechanism

can be established as:

$$\vec{r}_1 + \vec{r}_4 = \vec{r}_2 + \vec{r}_3 \tag{2}$$

Equation (2) can be expressed in polar notation as:

$$r_1 e^{j\theta_1} + r_4 e^{j\theta_4} = r_2 e^{j\theta_2} + r_3 e^{j\theta_3} \tag{3}$$

After applying Euler's equation to (3), the real and imaginary parts are:

$$\begin{aligned} r_1 cos\theta_1 + r_4 cos\theta_4 &= r_2 cos\theta_2 + r_3 cos\theta_3 \\ r_1 sin\theta_1 + r_4 sin\theta_4 &= r_2 sin\theta_2 + r_3 sin\theta_3 \end{aligned} \tag{4}$$

Left side of equation system (4) can be expressed in terms of θ_4 to obtain the angular position θ_3:

$$\begin{aligned} r_4 cos\theta_4 &= r_2 cos\theta_2 + r_3 cos\theta_3 - r_1 cos\theta_1 \\ r_4 sin\theta_4 &= r_2 sin\theta_2 + r_3 sin\theta_3 - r_1 sin\theta_1 \end{aligned} \tag{5}$$

The compact form of Freudenstein's equation is obtained by squaring the system (5) and adding its terms, as:

$$A_1 cos\theta_3 + B_1 sin\theta_3 + C_1 = 0 \tag{6}$$

with:

$$A_1 = 2r_3\left(r_2 cos\theta_2 - r_1 cos\theta_1\right) \tag{7}$$
$$B_1 = 2r_3\left(r_2 sin\theta_2 - r_1 sin\theta_1\right) \tag{8}$$
$$C_1 = r_1^2 + r_2^2 + r_3^2 - r_4^2 - 2r_1 r_2 cos\left(\theta_1 - \theta_2\right) \tag{9}$$

θ_3 can be obtained as a function of the parameters A_1, B_1, C_1 and θ_2, expressing $sin\theta_3$ and $cos\theta_3$ in terms of $tan\left(\theta_3/2\right)$ as follows:

$$sin\theta_3 = \frac{2tan(\theta_3/2)}{1+tan^2(\theta_3/2)} \quad , \quad cos\theta_3 = \frac{1-tan^2(\theta_3/2)}{1+tan^2(\theta_3/2)} \tag{10}$$

A second-order lineal equation is obtained by substitution on (6):

$$\left[C_1 - A_1\right] tan^2\left(\frac{\theta_3}{2}\right) + \left[2B_1\right] tan\left(\frac{\theta_3}{2}\right) + A_1 + C_1 = 0 \tag{11}$$

Solving (11), the angular position θ_3 is given by:

$$\theta_3 = 2arctan\left[\frac{-B_1 \pm \sqrt{B_1^2 + A_1^2 - C_1^2}}{C_1 - A_1}\right] \tag{12}$$

B. Kinematics of the coupler

Since C is the point of interest in the coupler, to determine its position in the system OX_rY_r it has to be established that:

$$\begin{aligned} C_{xr} &= r_2 cos\theta_2 + r_{cx} cos\theta_3 - r_{cy} sin\theta_3 \\ C_{yr} &= r_2 sin\theta_2 + r_{cx} sin\theta_3 + r_{cy} cos\theta_3 \end{aligned} \tag{13}$$

Translated to the global coordinate system, this point is expressed as:

$$\begin{bmatrix} C_x \\ C_y \end{bmatrix} = \begin{bmatrix} cos\theta_0 & -sin\theta_0 \\ sin\theta_0 & cos\theta_0 \end{bmatrix} \begin{bmatrix} C_{xr} \\ C_{yr} \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{14}$$

As can be observed, equations (13), (14) and the expressions corresponding to mechanism kinematics are sufficient to calculate the position of C along the trajectory.

V. Case Study

The problem addressed in this work is the length synthesis of a four-bar mechanism, designed for tracking a vertical linear trajectory indicated by a set of N precision points, without prescribed synchronization; that is, the point C of the coupler must *pass over* every point consecutively, without a pre-established sequence for such positions. The case study was selected because of its complexity; a measure for this complexity is the parameter ρ, which stands for the ratio between the feasible zone and the search space and can be represented by the percentage of feasible solutions found in an arbitrarily large set of randomly-generated vectors [34]. As the value of ρ diminishes, the computational effort required by the solving algorithm increases, since there are fewer available solutions. In order to evaluate ρ for this case a million of proposed solutions were taken, and only forty-three of them were feasible, resulting in a value of $\rho = 0.0043\%$.

Without loss of generality, the constrained optimization problem can be defined as to:

$$minimize \quad f(\vec{x}) \tag{15}$$

subject to:

$$g_j(\vec{x}) \leq 0, \qquad j = 1,2,...,p \tag{16}$$

$$h_k(\vec{x}) = 0, \qquad k = 1,2,...,q \tag{17}$$

where $\vec{x}$ is the vector of variables with dimension n, $f(\vec{x})$ is the objective function, $g_j(\vec{x})$ is the set of p inequality constraints and $h_k(\vec{x})$ is the set of q equality constraints.

A. Objective function

As a result of the mechanism synthesis will be calculated the length of the bars, the distance and rotation angle between the coordinate systems, and the angle set for the input bar. In the global coordinate system OXY, the precision point C_d^i is indicated as:

$$C_d^i = \left[C_{xd}^i, C_{yd}^i\right]^T \tag{18}$$

The set of precision points is defined as:

$$\Omega = \left\{C_d^i | i \in N\right\} \tag{19}$$

Given a set of values of the mechanism bars and their parameters x_0, y_0, θ_0, each position of the coupler can be expressed as a function of the input bar angle:

$$C^i = \left[C_x(\theta_2^i), C_y(\theta_2^i)\right]^T \tag{20}$$

It is desired to minimize the distance between the calculated and the precision points, C^i and C_d^i respectively; the function in (21) is proposed to quantify this error:

$$f(\theta_2^i) = \sum_{i=1}^{N} \left[(C_{xd}^i - C_x^i)^2 + (C_{yd}^i - C_y^i)^2\right] \tag{21}$$

B. Design constraints

The fulfillment of the performance constraints related to dimensions and mobility criteria of the mechanism is fundamental in its design, since these limits establish the physical reproducibility and esthetic.

1) Sequence of input angles: The generation of a trajectory without prescribed synchronization requires an ascendant or descendant order of the crank angle values, in correspondence to each precision point. If the angle for the precision point i is denoted as θ_2^i, this order can be expressed as:

$$\theta_2^1 < \theta_2^2 < ... < \theta_2^N \tag{22}$$

where N is the number of precision points.

2) Grashof's law: It is a fundamental consideration of design, since it defines the criteria to ensure complete mobility for at least one link on a four-bar mechanism. This law establishes that *for a planar four-bar linkage, the sum of the shortest bar and the largest bar cannot be larger than the sum of the remaining bars, if a continual relative rotation between two elements is desired* [12]. If the lengths of the shortest and largest links are denoted as s and l respectively, with p and q indicating the remaining links, it is established that:

$$l + s \leq p + q \tag{23}$$

For this synthesis problem, Grashof's law is given by:

$$r_1 + r_2 \leq r_3 + r_4 \tag{24}$$

Consequently the following restrictions are required:

$$r_2 < r_3, \quad r_3 < r_4, \quad r_4 < r_1 \tag{25}$$

C. Design variables

Consider the vector of design variables for the four-bar mechanism, established as:

$$\begin{aligned} \vec{p} &= [p_1, p_2, ..., p_{15}]^T & (26)\\ &= \left[r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, ..., \theta_2^6\right]^T & (27) \end{aligned}$$

where the first four variables are the mechanism bar lengths, the following two values represent the length of the supporting bars of the coupler, the subsequent three variables indicate the relative position between the coordinate systems, and the last six values correspond to the angle sequence for the input bar.

D. Optimization problem

Consider the mono-objective numerical optimization problem described by (28) to (41), to obtain the dimensional synthesis of a four-bar mechanism for trajectory generation over N precision points:

$$min \quad f(\vec{p}) = \sum_{i=1}^{N} \left[(C_{xd}^i - C_x^i)^2 + (C_{yd}^i - C_y^i)^2\right] \tag{28}$$

subject to:

$$\begin{aligned} g_1(\vec{p}) &= p_1 + p_2 - p_3 - p_4 \leq 0 & (29)\\ g_2(\vec{p}) &= p_2 - p_3 \leq 0 & (30)\\ g_3(\vec{p}) &= p_3 - p_4 \leq 0 & (31)\\ g_4(\vec{p}) &= p_4 - p_1 \leq 0 & (32)\\ g_5(\vec{p}) &= p_{10} - p_{11} \leq 0 & (33)\\ g_6(\vec{p}) &= p_{11} - p_{12} \leq 0 & (34)\\ g_7(\vec{p}) &= p_{12} - p_{13} \leq 0 & (35)\\ g_8(\vec{p}) &= p_{13} - p_{14} \leq 0 & (36)\\ g_9(\vec{p}) &= p_{14} - p_{15} \leq 0 & (37) \end{aligned}$$

and the precision points:

$$\Omega = \{(20,20),(20,25),(20,30),(20,35),(20,40),(20,45)\} \tag{38}$$

with the bounds:

$$\begin{aligned} 0 &\leq p_i \leq 60, & i = [1,2,3,4] \quad (39)\\ -60 &\leq p_i \leq 60, & i = [5,6,8,9] \quad (40)\\ 0 &\leq p_i \leq 2\pi, & i = [7,10,11,...,14,15] \quad (41) \end{aligned}$$

VI. Optimization Algorithm

Algorithms 1 and 2 correspond to the global and local search sections of the proposed memetic MemMABC, respectively. This algorithm requires six user-defined parameters: the number of sources or possible solutions SN, the maximum number of cycles or generations MCN, the maximum number of the objective function evaluations $MaxEvs$, the frequency of LS activation $Frequency$, the LS depth $MaxCount$, and the number of consecutive trials for improvement a source is kept before being replaced $TryLimit$, that is calculated as:

$$TryLimit = SN * n \tag{42}$$

where n is the number of design variables.

The implementation of the proposed algorithm was programmed in MATLAB R2013a, and the simulations were carried out on a computational platform with the following characteristics: Intel Core i7@2.6 GHZ microprocessor, 8Gb RAM memory and Windows 8 Operating system. All the algorithm simulations were executed with the following parameter values: $SN = 50$, $MCN = 8,000$, $MaxEvs = 1,000,000$, $Frequency = 1,750$, $MaxCount = 40,000$, and $TryLimit = 750$, calculated from (42).

VII. Analysis of Results

Thirty independent runs were executed for the selected case study with both algorithms, MABC and MemMABC; a statistical analysis of their results is presented in Table I. As can be seen, the minimum value for the objective function was obtained with MemMABC ($OF = 0.014667757429261$), with a variance of $\sigma^2 = 0.086822826371684$, significantly lesser than the correspondent evaluation for MABC; since the variance measures the dispersion of a set of random variables with respect to their arithmetic mean, this value indicates a steady operation of the algorithm. Additionally, MemMABC required a 10% less evaluations to find its best result, even before reaching the stop condition given by $MaxEvs$. The results show the synergy formed by the combination of these global and local searchers, and the inclusion of a technique for handling of constraints.

TABLE I
Statistical Analysis of MABC vs MemABC

Parameter	**MABC**	**MemMABC**
Minimum	0.029598038968931	0.014667757429261
Maximum	6.047393204762160	1.324383428475970
Variance	1.158659308687650	0.086822826371684
Standard Dev	1.076410381168650	0.294657133583566
Evaluations Req	1,000,000	900,000

Figure 3 shows a simulation of the best-solution mechanism calculated by MemMABC, and its trajectory over the precision points, marked as $C_1, C_2, ..., C_6$. As it can be seen, the mechanism passes over the precision points in its ascending path and the return loop is quite small. Consequently the recovering time is short, a plus if the device is analyzed from an engineering point of view.

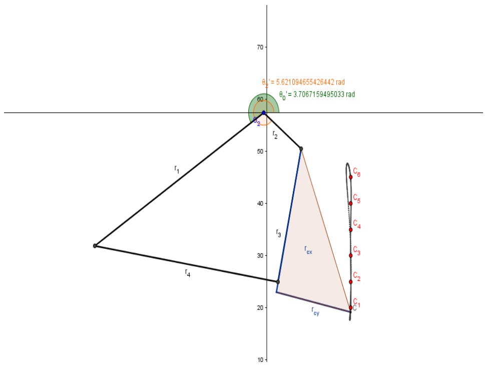

Fig. 3. Simulation of MemMABC best solution

A set with five solutions was selected considering the best values of the objective function; Table II present the solution vectors included in that set. As can be seen, all the values fell within the limits marked by design constraints; because of the table size, quantities are represented using only five decimal digits in spite of being calculated in the simulation with a precision of fourteen decimal places. The results generated after the simulations demonstrate the capability of MemMABC for balancing diversification over the area of feasible solutions and intensification for local exploitation.

TABLE II
Best five solution vectors

Variable	1	2	3	4	5
r_1	47.61022	47.65784	50.59788	50.10518	37.74315
r_2	11.29351	10.11871	12.73210	10.64155	11.63338
r_3	26.05979	25.87023	23.19406	26.15958	23.60134
r_4	44.17324	42.22803	47.15120	43.20891	37.09528
r_{cx}	28.09277	26.33869	21.27262	24.18547	28.33434
r_{cy}	18.09717	20.42422	15.87719	20.65887	7.90358
θ_0	3.70672	3.70295	3.57266	3.66212	3.80425
x_0	-0.72692	-1.06722	4.14226	0.71014	5.86156
y_0	57.39457	57.74773	53.35509	57.35837	58.09489
θ_2^1	2.03045	1.70228	2.09284	1.77039	1.02760
θ_2^2	2.54567	2.44986	2.55090	2.44951	2.14131
θ_2^3	2.97677	2.93643	2.94565	2.91994	2.58235
θ_2^4	3.40864	3.40469	3.35006	3.37768	3.02345
θ_2^5	3.87709	3.90614	3.79183	3.87046	3.53605
θ_2^6	4.45365	4.55469	4.30554	4.50137	4.21593
OF	0.01467	0.02635	0.02968	0.03634	0.04452

VIII. Conclusion

A novel proposal of a memetic algorithm for solving real-world engineering design problems is presented in this paper, using a combination of algorithms with Modified Artificial Bee Colony and Random Walk. From the obtained results it is established that: 1) the algorithm improves the performance of MABC, not only in relation to the search for the optimal, but in reference to the stability of the search; 2) considering the point of view of engineering design, this optimization method produces good solutions since they are physically and esthetically reproducible; 3) no extensive computing resources are required for its implementation, and 4) it is a simple algorithm with an easy implementation. It should be mentioned that wide ranges of values for the design variables were used for simulating solutions of the proposed optimization problem, so results can be improved if such ranges are bounded more closely, accordingly to the physical specifications of the real model.

Although in this paper a specific case of synthesis for a four-bar mechanism is addressed, the simplicity of the proposed algorithm facilitates its use for the designing of other types of mechanisms and devices. In this sense, the main difficulty is an adequate interpretation and formulation of the particular problem and its corresponding constraints. The initial configuration of the algorithm requires special attention and is a line that can be the base for future work, considering both the previous tuning of parameters and their control during execution.

Finally, the main future work for this development is its transformation from a canonical memetic to an adaptive algorithm, with the capability to modify itself by incorporating knowledge to this process of self adaptation. This transformation implies the application of new techniques for local search, and to implement the intelligence required to process more than one meme for the learning.

Acknowledgment

The authors would like to thank Instituto Politécnico Nacional (IPN) for its support via Secretaría de Investigación y Posgrado (SIP) with the research project SIP-20151320.

References

[1] I. Boussaid, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Information Sciences*, vol. 237, pp. 82–117, 2013.

[2] X. Yang, *Harmony Search as a Metaheuristic Algorithm*. Springer Berlin, 2009, vol. 191, pp. 1–14.

[3] S. Domínguez Isidro, E. Mezura Montes, and G. Leguizamón, "Memetic differential evolution for constrained numerical optimization problems," in *Proceedings of 2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 2996–3003.

[4] N. Krasnogor, A. Aragón, and J. Pacheco, *Memetic Algorithms*. Springer Berlin, 2012, vol. 2, ch. 12, pp. 905–936.

[5] E. Ozcan and C. Basaran, "A case study of memetic algorithms for constraint optimization," *Soft Computing*, vol. 13, pp. 871–882, 2009.

[6] X. Cai, Z. Hu, and Z. Fan, "A novel memetic algorithm based on invasive weed optimization and differential evolution for constrained optimization," *Soft Computing*, no. 17, pp. 1893–1910, 2013.

[7] Y. Li, B. Wu, L. Jiao, and R. Liu, "Memetic algorithm with double mutation for numerical optimization," in *IScIDE 2011*, 2012, pp. 66–73.

[8] P. Garg, "A comparison between memetic algorithm and genetic algorithm for the cryptoanalysis of simplified data encryption standard algorithm," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 1, no. 1, pp. 33–42, April 2009.

[9] Y. Zhang, F. Gao, Y. Zhang, and W. A. Gruver, "Dimensional synthesis of a flexible gripper with a high degree of stability," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, October 1996, pp. 1025–1030.

[10] P. Merz and B. Freisleben, "Memetic algorithms for the traveling salesman problem," *Complex Systems*, no. 13, pp. 297–345, 2001.

[11] J. Ni, L. Li, F. Qiao, and Q. Wu, "A novel memetic algorithm and its application to data clustering," *Memetic Computation*, 2013.

[12] J. E. Shigley and J. J. Uicker, *Teoría de Máquinas y Mecanismos*. México: McGraw Hill, 1988.

[13] R. Norton, *Diseño de Maquinaria, una Introducción a la Síntesis y Análisis de Mecanismos y Máquinas*. México: McGraw Hill, 1995.

[14] P. Moscato and C. Cotta, "Una introducción a los algoritmos meméticos," *Revista Iberoamericana de Inteligencia Artificial*, vol. 19, pp. 131–148, 2003.

[15] P. A. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," Caltech, Technical Report 826, 1989.

[16] F. Neri and C. Cotta, *Handbook of Memetic Algorithms*. Springer Verlag, 2012, ch. A Primer on Memetic Algorithms.

[17] Y. Ong, M. Lim, and X. Chen, "Memetic computation: Past, present and future," *IEEE Computational Inteligence Magazine*, vol. 5, no. 2, 2010.

[18] X. Chen, Y. Ong, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 5, pp. 591–607, 2011.

[19] K. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: Model, taxonomy, and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474–488, October 2005.

[20] M. A. Montes de Oca, C. Cotta, and F. Neri, *Local Search*. Springer Verlag, 2012, vol. 379, pp. 29–42.

[21] J. Tang, M. Lim, and Y. Ong, "Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems," *Soft Computing*, vol. 11, no. 9, pp. 873–888, July 2007.

[22] S. D. P. Rakshit, A. Konar and A. Nagar, "ABC-TDQL: An adaptive memetic algorithm," in *Proceedings of 2013 IEEE Workshop on Hybrid Intelligent Models and Applications*, April 2013, pp. 35–42.

[23] ——, "A bee foraging-based memetic harmony search method," in *Proceedings of 2012 IEEE International Conference on Systems, Man and Cybernetics*, October 2012, pp. 184–189.

[24] I. J. Fister, X. Yang, I. Fister, and J. Brest, "Memetic firefly algorithm for combinatorial optimization," Research Gate (online), April 2012.

[25] P. Moscato and C. Cotta, *A Gentle Introduction to Memetic Algorithms*. Kluwer Academic Publishers, 2003, pp. 105–144.

[26] N. Z. Y.S. Ong, M.H. Lim and K. Wong, "Classification of adaptive memetic algorithms: A comparative study," *IEEE Transactions on Systems, Man and Cybernetics, Part B Cybernetics*, vol. 36, no. 1, pp. 141–152, February 2006.

[27] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Erciyes University, Technical Report TR06, 2005.

[28] E. Mezura Montes and O. Cetina Domínguez, "Empirical analysis of a modified artificial bee colony for constrained numerical optimization," *Applied Mathematics and Computation*, vol. 218, pp. 10 943–10 973, 2012.

[29] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, pp. 311–338, 2000.

[30] M. Mernik, S. Liu, D. Karaboga, and M. Crepinsek, "On clarifying misconceptions when comparing variants of the artificial bee colony algorithm by offering a new implementation," *Information Sciences*, vol. 291, pp. 115–127, 2015.

[31] S. Rao, *Engineering Optimization: Theory and Practice*, 4th ed. John Wiley and Sons, 2009.

[32] R. Pérez, *Análisis de Mecanismos y Problemas Resueltos*. Alfaomega Grupo Editor S.A. de C.V., 2006.

[33] E. Vega-Alvarado, E. Santiago-Valentín, A. Sánchez-Márquez, A. Solano-Palma, E. A. Portilla-Flores, and L. Flores-Pulido, "Síntesis óptima de un mecanismo plano para seguimiento de trayectoria utilizando evolución diferencial," *Research in Computing Science*, vol. 72, pp. 85–89, 2014.

[34] J. Liang, T. Runarsson, E. Mezura Montes, M. Clerc, P. Suganthan, C. Coello, and K. Deb, "Problem definitions and evaluation criteria for the CEC 2006," Nanyang Technological University, Technical Report, 2006.